Privacy preservation on categorical datasets through Minimal Infrequent Itemset suppression

J.S. ten Tusscher

Supervisors:

prof. dr. A.P.J.M. Siebes

dr. A.J. Feelders

June 13, 2024

Abstract

Minimal Infrequent Itemsets (MIIs) are infrequent itemsets that have no infrequent proper subsets. Different authors have explored the idea of detecting anomalies or privacy leaks in datasets by mining and analyzing MIIs. Because MIIs have been researched repeatedly in such contexts, it seems fitting to try and use MIIs for data sanitization. In this work, we explore this idea and develop and analyze different MII-based sanitization algorithms with various privacy guarantees like k-anonymity and (ϵ, δ)-differential privacy. Experimental results show that these algorithms yield sanitized datasets with good utility for different privacy parameters, datasets, and utility measures. These algorithms are meant for rectangular categorical $m \times n$ datasets, which is a common type of dataset in the world of privacy preservation. Because of our heavy use of MIIs in our algorithms, this begs the question how many MIIs a rectangular dataset can contain. We prove that a rectangular $m \times n$ dataset can contain at most $m\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Theta\left(m2^{n-\frac{\log(n)}{2}}\right)$ MIIs, and we prove that this bound is tight for all n and all infrequency thresholds θ as long as m is sufficiently large. This means that mining MIIs on a rectangular dataset requires exponential time and space with respect to n.

Contents

1 Introduction **3**

1.1 Background . . . 3

1.2 Introduction to privacy preservation . . . 4

1.3 Non-interactive Privacy Preserving Data Publishing . . . 6

1.3.1 k-anonymity . . . 6

1.3.2 ℓ-diversity . . . 8

1.3.3 t-closeness . . . 9

1.4 Interactive Privacy Preserving Data Publishing . . . 10

1.4.1 Differential privacy . . . 10

1.4.2 Randomized Response, Local differential privacy . . . 12

1.4.3 Non-interactive differential privacy . . . 13

1.5 Itemset mining . . . 13

1.5.1 (Maximal) frequent itemsets . . . 14

1.5.2 (minimal) infrequent itemsets . . . 14

1.6 Privacy Preserving Data Mining . . . 15

1.7 Utility . . . 16

2 MII-based privacy preservation **17**

2.1 Privacy through Imputation . . . 17

2.2 Mining and suppressing revealing data . . . 18

2.3 MII-SUPPRESS as a k-anonymity algorithm . . . 21

2.4 Towards Differential Privacy . . . 22

2.4.1 Differential Privacy under Sampling . . . 23

2.4.2 Parameter choices 26
2.4.3 Utility 27

3 Complexity analysis of MII mining on categorical datasets 29
3.1 Complexity of FI mining 29
3.2 Frequent and infrequent mining algorithms 30
3.2.1 Comparison of complexities of FI and MII mining 31
3.3 A tight bound on the number of MIIs in a categorical dataset 32
3.3.1 Proof of tightness using Orthogonal Arrays 34
3.4 ∞ 38
3.5 ∞ without the primes 39
3.6 Non-OA matrices that contain $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs 42
3.7 In summary 43

4 Experiments 45
4.1 Datasets, utility measures, implementation details 46
4.1.1 DP-MII-SUPPRESS 48
4.2 Results 48
4.2.1 MII-SUPPRESS and K-MII 48
4.2.2 DP-MII-SUPPRESS 50
4.3 Discussion 51
4.3.1 MII-SUPPRESS 51
4.3.2 MII-SUPPRESS VS K-MII 51
4.3.3 DP-MII-SUPPRESS 53

5 Conclusion 55

Appendices 57

A 58

Chapter 1

Introduction

1.1 Background

The starting point of this thesis was the idea of 'Privacy through Imputation'. Imputation generally refers to the task of filling in blanks in an incomplete dataset. However, we wanted to research if we can sanitize a complete dataset by making it incomplete and then performing imputation. To do this, we can first identify the parts of the dataset that are dangerous from a privacy perspective, then clear them out, and then impute them. Which parts of the dataset are dangerous is unclear because the word 'dangerous' itself is very ambiguous. We decided to focus on finding and imputing Minimal Infrequent Itemsets (MIIs), mainly for two reasons:

1. MIIs have been linked to tasks closely related to privacy preservation, but have to our knowledge never been used for the actual privacy preservation task itself. For example, multiple authors bring up the idea that MIIs can be used to detect to what extent a dataset was properly sanitized[19][18]. We want to take this idea one step further, and research if we can actually sanitize a dataset by mining, suppressing and imputing MIIs.

2. It seems to be unclear how many MIIs a rectangular dataset can contain. For example, Gupta et al. write[18]:

   > 'For some datasets, the set of infrequent itemsets can be exponentially large. [...] Hence, it is essential to report only the minimally infrequent itemsets.'

   This immediately begs the question how many MIIs a dataset can contain, but we could

not find a paper that answers this question for the type of datasets we have focused on (rectangular categorical datasets). Therefore we have analyzed this problem as well.

In the end, this thesis is about the complexity of MIIs as much as it is about data sanitization. Hence, in the remainder of this introductory chapter we will introduce the necessary concepts from data sanitization and itemset mining. In chapter 2 we develop MII based privacy preserving algorithms that revolve around the idea of Privacy through Imputation. In chapter 3 we analyze the theoretic time and space complexity of MII mining, and in chapter 4 we run experiments using the algorithms developed in chapter 2.

1.2 Introduction to privacy preservation

An $m \times n$ dataset D is comprised of rows and columns. Every row represents an entity, e.g. a company or person. Every column represents an attribute, e.g. `age`, `nationality` or `ZIP code`. If D contains sensitive information about people, e.g. medical information, then it is not wise to mindlessly publish the dataset. At the same time, it can actually be useful to share medical data about patients with hospitals, researchers and data analysts, because this can facilitate research that can eventually help patients. In order to allow this, altered data can be published such that useful patterns can be mined but privacy of the participants of the dataset is protected.

There are two main ways to achieve this: Privacy Preserving Data Publishing (PPDP), and Privacy Preserving Data Mining (PPDM). PPDP can be subdivided into two categories: non-interactive and interactive PPDP. In the case of non-interactive PPDP, the owner of the data, often called a *curator*, will sanitize the dataset. The resulting sanitized release, denoted by $\tilde{D}$ from this point onwards, can then be published. In the case of interactive PPDP, analysts can submit statistical queries to the curator. The curator will run the queries on D, sanitize the query output (usually by perturbing the output such that not too much information is released), and return the sanitized result to the analysts. Lastly, PPDM concerns itself with adapting data mining algorithms in such a way that the result of the mining algorithms does not leak sensitive information.

If used carelessly, both PPDP and PPDM can release poorly sanitized information, in which case an *attacker* may be able to learn things about people in the dataset that they

should not have been able to learn. This is called unintended *information disclosure*[13]. A distinction is commonly made between two types of information disclosure[24][13]:

1. identity disclosure, also called re-identification, where the identity of a person can be linked to an entry in the dataset;

2. attribute disclosure, where new information can be inferred about a person.

Note that identity disclosure on its own can be harmful, even when the dataset contains no additional attributes, for instance when your name appears on a list of resistance members during a time of oppression. At the same time, attribute disclosure can take place without identity disclosure; Lambert[13] gives the example where average salaries of unionized workers are published, and you are such a unionized worker. Even though you as a person are not directly part of the dataset, an adversary can still learn something about you if they know you are a unionized worker.

How troubling attribute disclosure is, depends in part on the type of attribute. Roughly speaking, data attributes are subdivided into three distinct categories[37]:

1. **Identiers**, e.g. someone's full name or social security number. It is very common to remove all identifiers from the dataset completely[37][34], because they instantly lead to identity disclosure.

2. **Sensitive attributes**, e.g. a medical diagnosis, religious beliefs.

3. **Quasi-identifiers** (QIDs), a term coined in 1986 by Dalenius[10]. These are attributes that do not identify individuals on their own, but when combined can still uniquely identify them.

Whereas identifiers are often removed completely, this is not the case for sensitive attributes. After all, it is useful to release sensitive information as long as a certain amount of privacy can be guaranteed. Furthermore, note that in theory an attribute can be both a sensitive attribute *and* a quasi-identifier. For example, if you know about your friends medical history, then you may be able to re-identify them in a medical dataset based on medical data alone. In other words: the medical attributes are both quasi-identifiers *and* sensitive attributes in this case.

We will now give some examples of the problematic nature of quasi-identifiers. `ZIP code`, `gender` and `date of birth` are all QIDs that do not point to unique individuals on their own, but when they are combined they uniquely identify 87% of the United States population[36]. Therefore, if an attacker knows someone's `ZIP code`, `gender` and `date of birth`, and only one entry in the dataset has these exact values, then the dataset discloses the identity of this person with high probability. Furthermore, if the dataset contains additional (sensitive) attributes that are not public information, then the dataset also causes sensitive attribute disclosure and teaches the attacker new things about someone despite no explicit identifiers being present anywhere in the dataset.

Another example of this is a 2013 article in Nature that examines whether anonymous mobile phone location data pinpoints individuals. The paper concludes that "in a dataset where the location of an individual is specified hourly, and with a spatial resolution equal to that given by the carrier's antennas, four spatio-temporal points are enough to uniquely identify 95% of the individuals"[11].

Now that we have developed a bit of a feel for how seemingly harmless data can still lead to identity disclosure and attribute disclosure, we will look at how PPDP and PPDM can be used to protect against this.

1.3 Non-interactive Privacy Preserving Data Publishing

Many different privacy preservation approaches and algorithms have been developed in the past few decades. Some algorithms work for specific types of data – think of an algorithm specifically developed for sanitizing sensitive time series data[3], or an algorithm that anonymizes pictures with faces in them without changing the underlying data distribution, by using a generative adversarial neural network to generate a new face to overlay the picture with[21]. However, many approaches focus on sanitizing tables, i.e. $m \times n$ datasets, the most well known one being k-anonymity.

1.3.1 k-anonymity

k-anonymity was proposed in 1998 by Samarati and Sweeney[34]. If a release has k-anonymity, then any combination of quasi-identifiers present in a row is present in at least $k-1$ other

rows. Such a set of $\geq k$ rows forms an equivalence class. If a combination of quasi-identifying values does not conform to k-anonymity prior to anonymization, then values in D can either be suppressed, i.e. the cells holding the quasi-identifiers will be cleared out, or generalized, e.g. exact ages are replaced with age intervals, such that each interval occurs at least k times. Strings, such as first names, could be generalized by shortening them such that any shortened name occurs at least k times. One could theoretically do something similar with fully identifying strings, but it is common for a k-anonymity algorithm to not bother and remove the identifiers completely.[34].

first	**last**	**age**	**race**
Harry	Stone	34	Afr-Am
John	Reyser	36	Cauc
Beatrice	Stone	47	Afr-Am
John	Ramos	22	Hisp

first	**last**	**age**	**race**
*	Stone	30-50	Afr-Am
John	R*	20-40	*
*	Stone	30-50	Afr-Am
John	R*	20-40	*

Table 1.1: The left table represents a simple dataset. The right table is the same dataset, but satisfies k-anonymity for $k = 2$. These tables were taken from [31].

An example of k-anonymity can be found in table 1.1, with D on the left and $\tilde{D}$ on the right. An asterisk means that the cell was suppressed. Note that in $\tilde{D}$ no combination of attribute values exists that points to fewer than 2 entries, which is why the dataset on the right is 2-anonymous. Note that usually, in the context of algorithms that yield a sanitized release, a dataset also has at least one sensitive attribute but this is not the case in this example.

Meyerson and Williams proved in 2004[31] that achieving optimal k-anonymity suppression – optimal meaning: the number of suppressed cells is minimized such that information loss is minimized – is NP-hard, but they proposed a polynomial algorithm with an approximation ratio of $O(k \log k)$. In practice this algorithm modified at most $4k \log k$ times the number of cells that would have been modified with optimal k-anonymity. Practical algorithms that find optimal k-anonymity were found later. For example, Bayardo and Agrawal approached the problem as a power set search problem[4], which is a common approach for itemset mining problems. Although their algorithm has exponential time complexity (as do all optimal k-anonymity algorithms due to the NP nature of the problem), it performed well on real world

census data.

Homogeneity attack, background knowledge attack

k-anonymity protects well against identity disclosure[39], but if not used cautiously, it can yield a sanitized release that is actually prone to multiple types of attacks, two of which will be discussed next.

Imagine that an attacker has access to an sanitized dataset $\tilde{D}$ that has 1 sensitive field, blood type, and they know that a person p is present in the dataset and must be represented by one of k rows in a certain equivalence class in $\tilde{D}$. If all k rows have the blood type AB, then then the attacker knows that p has blood type AB, even though $\tilde{D}$ satisfies k-anonymity. This is the homogeneity attack. It was first described by Ohrn et al. in 1999[32].

Another attack, the background knowledge attack, is somewhat similar to the homogeneity attack. With this attack however, instead of all k rows having the same sensitive attribute value, they have different values but the attacker can use background knowledge in order to deduce which sensitive value (likely) belongs to person p. For example, if the sensitive field represents the value on an investment portfolio, and an attacker knows that person p is exceptionally rich and an active investor, then the attacker can find out with high probability what p's portfolio value is. After all, it is probable that only one of the k rows has an exceptionally high portfolio value, and this value therefore presumably points to p.

1.3.2 ℓ-diversity

As a response to the aforementioned attacks that k-anonymity is susceptible to, ℓ-diversity was proposed by Machanavajjhala et al.[30]. Similarly to k-anonymity, the goal of ℓ-diversity is to create equivalence classes by suppressing or generalizing values, such that every equivalence class in $\tilde{D}$ has a certain degree of generality. An equivalence class in a sanitized release is ℓ-diverse, where $\ell \in \mathbb{R}$ and $\ell > 1$, if it contains at least ℓ well-represented values for a sensitive attribute. Multiple ideas are proposed for when a value is well-represented, but most notable is *entropy ℓ-diversity*, where the entropy of the sensitive attribute should be at least $\log(\ell)$ in

any equivalence class:

$$\forall b \in B : -\sum_{s \in S} P(s,b) \log\big(P(s,b)\big) \geq \log(\ell)$$

where B is the set of all equivalence classes in $\tilde{D}$, S is the domain of the sensitive attribute, and $P(s,b) = \frac{|\{r \in b | s \in r\}|}{|b|}$ is the relative frequency of s in b. This measure is called entropy ℓ-diversity because the left hand side of the inequality is simply the formulation of information theoretic entropy. Note that this definition of ℓ-diversity implies that every equivalence class should contain at least ℓ unique sensitive values, making the homogeneity-attack impossible on an ℓ-diverse dataset. The background knowledge attack also becomes less feasible: ℓ-diversity favors equivalence classes with more uniformly distributed values for the sensitive attribute (after all, entropy increases when a probability distribution becomes increasingly uniform), so the aforementioned hypothetical background knowledge attack on a rich friend with a valuable investment portfolio becomes more difficult.

Skewness attack

Attacks that an ℓ-diverse dataset can be susceptible to were described by Li et al.[26]. One such attack is the *skewness attack*. The idea here is that if the distribution of a sensitive attribute is heavily skewed, then an ℓ-diverse equivalence class with a very different sensitive attribute distribution can give away a lot of information. Say we have an ℓ-diverse dataset, where the sensitive attribute is a boolean that represents the result of a certain medical test. This test result is positive for only 1% of the population. If an attacker knows that a certain individual is part of a particular equivalence class in which the sensitive attribute is negative only 50% of the time, then the attacker has learned a lot: the probability that the individual had a positive test attribute increased from 1% to 50%. Therefore, ℓ-diversity can still lead to attribute disclosure with high probability.

1.3.3 t-closeness

After describing certain attacks that work on ℓ-diversity, including the skewness attack, Li et al. develop a new method: t-closeness[26]. A dataset D has t-closeness if, for every equivalence class in D, the distance between the probability distribution of the sensitive attribute in that

equivalence class, and the probability distribution of the sensitive attribute in all of D is at most t. This upper-bound on what the distance can be makes the skewness attack infeasible, assuming that a reasonable value for t and a reasonable distance metric were used.

1.4 Interactive Privacy Preserving Data Publishing

In the previous section we discussed different non-interactive Privacy Preserving Data Publishing approaches, where the goal is to release an sanitized version of the dataset. In contrast, interactive Privacy Preserving Data Publishing (sometimes called Privacy Preserving Data Analysis) revolves around the idea that an analyst can submit queries that can be executed against the original dataset, and the query result is then returned to the analyst. These queries do not return parts of the dataset, but only return statistical aggregate information, e.g. the ratio healthy to non-healthy individuals, or the relative number of young men who have been involved in at least one car crash.

Reconstruction attack

One might think that by only returning aggregate information, we automatically protect the privacy of the entities in the dataset. However, it has been shown that aggregate information, even perturbed aggregate information, can be used to reconstruct (parts of) the original dataset. Such an attack is called a *reconstruction attack*. The first reconstruction attack came from Dinur and Nissim[12], who showed that a simple binary string could be reconstructed by running a polynomial number of queries that each returned somewhat perturbed aggregate information.

Dwork and Roth later concisely summarized these findings in the Fundamental Law of Information Recovery: "'overly accurate' answers to 'too many' questions will destroy privacy in a spectacular way"[17]. In order to resolve this problem, differential privacy was developed.

1.4.1 Differential privacy

Differential privacy is the most popular interactive Privacy Preserving Data Publishing approach, although we should note that differential privacy can actually be applied to non-interactive PPDP as well. The core idea behind differential privacy, developed by Dwork et

al.[16][14], is that no query result should be significantly dependent on a single entry in the dataset. Because of this, not a lot of risk is involved in being part of the dataset. Differential privacy was developed in the years after the k-anonymity and its successors, and is these days considered to be the state of the art and go-to definition of privacy.

Differential privacy is defined as follows. Let $\mathcal{K}$ be a randomised function that is executed by the data curator before releasing any information. $\mathcal{K}$ provides ϵ-*differential privacy* if for all pairs of datasets D_1, D_2 that differ in exactly one row, and for all $\mathcal{S} \subseteq \text{im}(\mathcal{K})$, we have:

$$\mathbb{P}\big[\mathcal{K}(D_1) \in \mathcal{S}\big] \leq e^{\epsilon}\mathbb{P}\big[\mathcal{K}(D_2) \in \mathcal{S}\big]$$

where ϵ is a non-negative number that represents the amount of privacy loss. Generally, as the authors state in [14], we should consider ϵ to have a value like 0.1 ($e^{0.1} \approx 1.11$) or 0.01 ($e^{0.01} \approx 1.01$), although the authors also mention that values like $\ln 2$ or $\ln 3$ can be used occasionally. Which value is the 'right' value should be decided for each data set individually. Randomness is incorporated in the definition of differential privacy, because it can be proved that any privacy guarantee must depend on randomization[17]; without randomness, (partial) reconstruction attacks would always be possible.

Differential privacy offers multiple additional guarantees by definition, e.g.[17]:

1. Parallel composability: if $\mathcal{K}$ offers ϵ-differential privacy, and we partition dataset D into n disjoint non-empty datasets $D_1, D_2, \ldots, D_n$, then releasing $\mathcal{K}(D_1), \mathcal{K}(D_2), \ldots, \mathcal{K}(D_n)$ still offers ϵ-differential privacy. Similarly, if n different algorithms $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_n$ offer respectively $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$-differential privacy, and we partition D into $D_1, D_2, \ldots, D_n$, then releasing $\mathcal{K}_1(D_1), \mathcal{K}_2(D_2), \ldots, \mathcal{K}_n(D_n)$ gives $\max(\epsilon_1, \epsilon_2, \ldots, \epsilon_n)$-differential privacy.

2. Sequential composability: if n different algorithms $\mathcal{K}_1, \mathcal{K}_2, \ldots, \mathcal{K}_n$ offer respectively $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$-differential privacy, then releasing the results $\mathcal{K}_1(D)$, $\mathcal{K}_2(D)$, $\ldots$, $\mathcal{K}_n(D)$ gives $\left(\sum_{i=1}^{n} \epsilon_i\right)$-differential privacy.

3. Post-processing immunity: any function executed on ϵ-differentially private data cannot decrease the amount of privacy as long as it does not use auxiliary information.

In practice, ϵ-differential privacy is often too strong of a privacy guarantee, and therefore

a relaxed version of differential privacy can be used, called (ϵ, δ)-differential privacy:

$$\mathbb{P}\big[\mathcal{K}(D_1) \in \mathcal{S}\big] \leq e^{\epsilon}\mathbb{P}\big[\mathcal{K}(D_2) \in \mathcal{S}\big] + \delta$$

It should be noted that differential privacy simply quantifies the amount of privacy that is given up every time $\mathcal{K}$ is executed. This means that attackers can still learn a lot if they are allowed to run an arbitrary number of queries. Therefore, the curator should give every data analyst a *privacy budget* and if the collective privacy leakage incurred by the queries submitted by an analyst reaches the budget, the analyst is not allowed to submit any additional queries. This also shows one of the shortcomings of differential privacy: different analysts can cooperate and collectively go over their individual privacy budgets. This can be resolved by using one single budget that is shared between all analysts, but that means that if an analyst is late to the party, they cannot submit any queries. Another shortcoming of differential privacy, noted in recent years by Dwork, is that it is difficult for data curators to decide on the 'right' value for ϵ (and δ if applicable)[15].

1.4.2 Randomized Response, Local differential privacy

Above, the most common definition of differential privacy was given. This definition is sometimes called *global* differential privacy. The reason for this is that truthful information from individuals is collected by the data curator, and when a query is submitted by an analyst noise is added at a 'global' level to the query result in order to protect individual privacy. However, what if people do not trust the central data collecting authority? And what if the data curator *is* the analyst? For such cases, local differential privacy was developed, which is based on global differential privacy and the concept of randomized response.

Randomized response was proposed by Warner in 1965[40]. As a basic example (inspired by [17]), let Q be a sensitive yes/no question. A researcher R wants to know the yes/no ratio for this question at a population level, so R will interview a representative sample of size S, but people in the sample are afraid to answer Q truthfully because of its sensitive nature. Therefore, R asks Q to every person P, but before letting P answer, R instructs P to toss a fair coin. If the result is heads, P should answer Q truthfully. If the result is tails, P should toss the coin again, and say 'yes' in case of heads and 'no' in case of tails. R then leaves the room so they cannot see the result of the coin toss(es), and then returns a moment later to take

note of the answer P gives. R does this for all people in the sample and gets Y yes answers and N no answers, where $Y + N = S$. R knows that they got approximately $\frac{S}{4}$ random yes answers and approximately $\frac{S}{4}$ random no answers. Therefore, at population level the true ratio $\frac{yes}{no} \approx \frac{Y-\frac{S}{4}}{N-\frac{S}{4}}$. This way, randomized response allows people to answer sensitive questions with plausible deniability, while still allowing a data analyst to learn real information at a group level.

This basic idea of randomized responses has been used by Kasiviswanathan et al.[22] for the development *local* differential privacy. More formally, $\mathcal{K}$ provides ϵ-*differential privacy* if for all pairs of private user data u_1, u_2, and for all $\mathcal{S} \subseteq \text{im}(\mathcal{K})$, we have:

$$\mathbb{P}\big[\mathcal{K}(u_1) \in \mathcal{S}\big] \leq e^{\epsilon}\mathbb{P}\big[\mathcal{K}(u_2) \in \mathcal{S}\big]$$

As can be seen, with local differential privacy noise is added to the data of individuals before that data is collected by the central authority. The central authority can then collect the noisy data and still learn real information from it, without ever collecting the original non-noisy data at individual level. However, this does come at a cost: ensuring privacy *before* data collection will yield significantly lower utility in most cases.

1.4.3 Non-interactive differential privacy

Although differential privacy was mainly developed as an interactive PPDP approach, and most publications discuss it in such a context, it can also be applied to non-interactive PPDP. In fact, local differential privacy is an example of exactly that: the data curator collects noisy data from individuals and ends up with a noisy dataset that already satisfies differential privacy. The curator could therefore publish this dataset as-is while offering a differential privacy guarantee to the individuals in the dataset.

1.5 Itemset mining

In the previous sections we have discussed the most important concepts from data sanitization and privacy preservation. We will now briefly introduce concepts from itemset mining that we use in later chapters.

1.5.1 (Maximal) frequent itemsets

Say we have a dataset D. Following conventions from itemset mining, we call every row in D a *transaction*, and every value in every transaction an *item*. The set of all items in D is denoted by $\mathcal{I}$. A set $X \subseteq \mathcal{I}$ is called an *itemset*. We can calculate properties of X in D, such as X's support:

$$sup_D(X) = |\{t \in D \mid X \subseteq t\}|$$

The most well known itemset mining task is that of *frequent itemset mining*[1], which boils down to mining every itemset X in D for which $sup_D(X) \geq \theta$, where θ is the minimum support threshold. We let $\mathcal{F}_{FI}$ denote the resulting family of all frequent itemsets in D. This family has the downward closure property:

$$X \in \mathcal{F}_{FI} \wedge Y \subseteq X \Rightarrow Y \in \mathcal{F}_{FI}$$

This property makes sense intuitively: if $\{1, 2\}$ appears x times, then both $\{1\}$ and $\{2\}$ *must* also appear at least x times. Because of the downward closure property, rather than mining all frequent itemsets, we can also mine all *maximal* frequent itemsets[5]. An itemset X is a maximal frequent itemset iff $sup_D(X) \geq \theta$ and $\forall Y : Y \supset X : sup_D(Y) < \theta$. From the family of all maximal frequent itemsets, we can then compute *all* frequent itemsets thanks to the downward closure property. Because of this, we can achieve an exponential reduction in space in extreme cases. For example, say every item in an $m \times n$ dataset has support 1, which implies that $|\mathcal{I}| = mn$. If $\theta = 1$, then the family of all frequent itemsets will be $\{x \subseteq t, x \neq \emptyset \mid t \in D\}$, whereas the family of all maximal frequent itemsets will simply be $\{t \in D\}$. The respective sizes of these sets will be $m(2^n - 1)$ and m, so the former set is larger by a factor of $2^n - 1$.

It should be noted that in the context of privacy preservation, D is often assumed to be $m \times n$, i.e. rectangular, whereas this is not a necessary requirement in the context of itemset mining.

1.5.2 (minimal) infrequent itemsets

In the context of privacy preservation, we want to perform the complement task to frequent itemset mining: infrequent itemset mining. This task is clearly closely related to frequent

itemset mining, but has gotten less attention. Because infrequent itemset mining is the complement task to frequent itemset mining, there are many parallels here, as can be seen in the table below.

Frequent itemset (FI) mining	**Infrequent itemset (II) mining**
Mine X if $sup_D(X) \geq \theta$	Mine X if $sup_D(X) \leq \theta$
Maximal FI (MFI): $sup_D(X) \geq \theta \wedge \forall Y \supset X : sup_D(Y) < \theta$	Minimal II (MII): $sup_D(X) \leq \theta \wedge \forall Y \subset X : sup_D(Y) > \theta$
Downward closure: every FI is a subset of an MFI	Upward closure: every II is a superset of an MII

We want to note that in the context of frequent itemset mining, the term *negative border*, often denoted by $\mathcal{B}d^-$, is commonly used. The negative border of a dataset is in fact exactly the set of all MIIs.

To our knowledge, no paper on MIIs covers how many datasets an $m \times n$ dataset can contain, even though multiple papers tie MIIs to privacy preservation, and the most common data format in the context of privacy preservation is an $m \times n$ tabular dataset. Therefore we have researched this topic in chapter 3.

1.6 Privacy Preserving Data Mining

So far we have discussed interactive and non-interactive Privacy Preserving Data Publishing, but there is a third option: Privacy Preserving Data Mining (PPDM). Here, the goal is to perform a mining task on sensitive data in such a way that the result of the mining task can be published while protecting privacy.

In the context of PPDM, there are usually additional rules that need to be satisfied. One commonly studied topic is that of the *sensitive* frequent itemset. A sensitive frequent itemset is a frequent itemset that should not be returned by a PPDM mining task, either implicitly or explicitly. A question is how a transaction dataset D can be altered in such a way that sensitive frequent itemsets are hidden (and cannot be inferred with high probability from $\tilde{D}$), while supports of the remaining non-sensitive frequent itemsets are altered as little as possible. Multiple approaches have been proposed for this problem[33][35][29].

Even if we do not have a specific set of sensitive frequent itemsets, we may still want a general differential privacy guarantee when publishing the frequent itemsets from a dataset. One notable example here is Privbasis[28], which was inspired by work by Bhaskar et al.[7]. Privbasis does not blindly publish all itemsets with support $\geq \theta$. Instead, it constructs and publishes a differentially private θ-Basis Set. A θ-Basis Set is a set of itemsets, such that most frequent itemsets in D are a subset of a member of the θ-Basis Set. The θ-Basis Set is constructed stochastically in such a way that false positives and false negatives are possible, and a differentially private guarantee can be derived. Experiments have shown that when $\epsilon \geq 0.1$ the false negative and false positive rates are relatively low on well-known datasets.

1.7 Utility

In this chapter, we have discussed many of the popular approaches to publishing or mining data while preserving privacy. All these approaches focus on giving a privacy definition, and then quantifying privacy through one or more parameters like k, ℓ, t or ϵ. More extreme parameter values generally lead to more privacy preservation, but also to less useful data. This is commonly referred to as the *privacy-utility trade-off*. Rather than thoroughly deriving both a privacy guarantee and a utility guarantee, it is more common to only derive a privacy guarantee and then run experiments on real or synthetic datasets to explore the utility of a certain approach.

Utility can be measured in many different ways. In the context of non-interactive PPDP, it is very common to experiment with census data, usually the Adult dataset[6]. This is a binary classification dataset where the class label indicates whether an adult makes over $50,000 per month or not, so classification results from both D and $\tilde{D}$ are often compared. In the context of interactive PPDP on the other hand, it is difficult to quantify utility, either formally or through experiments, because it depends on the kind of queries the analyst submits to the data curator. Lastly, in the context of PPDM, it is customary to compare the privacy preserving mining result with the non privacy preserving mining result. For example, PrivBasis compares its approximation of the frequent itemsets with the real frequent itemsets.

Chapter 2

MII-based privacy preservation

2.1 Privacy through Imputation

This work started with the idea of Privacy through Imputation. Normally, imputation refers to identifying missing data and filling it in by extrapolating from non-missing data or background information. In the context of privacy however, we assume that the original dataset D is a complete dataset, so there are no missing values. Rather we will impute the values in D that contain *too much* information, because they cause a row in the dataset to 'stand out' which could pose a privacy risk.

Generally, missing data is imputed by sampling from the non-missing values in the dataset. However, this can quickly lead to problems in our case. After all, if our dataset D contains two sensitive values that we want to impute, named A and B, then we want to make sure that we do not accidentally impute A by sampling B from D. We could use a more complex sampling approach to handle this problem, but we choose to mitigate it in a simpler manner: we completely remove all dangerous parts, in this case all of A and all of B, before we perform the imputation step. This means that in the end, Privacy through Imputation becomes a three step task:

1. Identify the parts of D that are too sensitive
2. Remove these parts from D (i.e. suppress them completely in D.)
3. Impute the missing values.

Privacy through Imputation can be viewed as a subcategory of non-interactive Privacy Preserving Data Publishing, i.e. the goal is to publish an entire (sanitized) dataset $\tilde{D}$ that can then be published and analyzed extensively by anyone who gets access. The approach has two sources of privacy: the removal of sensitive parts, and the imputation which adds uncertainty to the dataset. In this chapter, we attempt to develop algorithms that implement this framework. We also investigate if such algorithms can give a differentially private guarantee, because that is considered the gold standard in this field.

2.2 Mining and suppressing revealing data

In the context of k-anonymity and similar algorithms, a distinction is made between identifiers, quasi-identifiers (QIDs), and sensitive attributes. The identifiers should always be removed for very obvious reasons, and therefore we assume that the columns that contain identifiers have already been removed from our dataset. QIDs and sensitive attributes remain, and basic k-anonymity only anonymizes QIDs. This can cause all sorts of problems, not in the least for the data curator because it can be unclear if a variable is a QID or a sensitive attribute or both, so we decide to simply include all variables in the sanitization procedure. This refinement of k-anonymity reduces the chance of human error by the curator and benefits privacy, and has actually already been suggested by other authors like in[27].

At this point we no longer have QIDs and sensitive attributes, we just have attributes. We can subdivide the attributes into numerical and categorical ones. Both have different properties and require different strategies. We decide to focus on categorical data, in part because numerical data can be converted into categorical data through binning.

This means that we end up with an $m \times n$ dataset D that is comprised of n categorical variables that each have a finite number of values. We can simply say that a value x should be suppressed and imputed if it is infrequent, i.e. for an infrequency threshold θ we impute x iff

$$sup_D(x) \leq \theta$$

This is not a sound privacy approach, because the joint probability of multiple frequent values can still be low and therefore too revealing. Therefore, we can generalize this approach and say that any infrequent itemset x should be imputed. Now the pendulum swings the other way,

because as discussed in chapter 1, every superset of an infrequent itemset must be infrequent due to upward closure, meaning that now every row will be imputed as a whole if it contains an infrequent subset, no matter how small this subset is. herefore, we decide to impute minimal infrequent itemsets (MIIs) because we think this strikes the privacy-utility balance best:

1. The family of MIIs serves as a generator for all infrequent itemsets, so the MIIs in a row form the core of what makes a revealing row revealing.

2. We expect the average MII to be rather small with respect to n, which should greatly improve utility over blindly imputing entire infrequent rows.

This means that every itemset x will be imputed for which

$$sup_D(x) \leq \theta \wedge \forall y \subset x : sup_D(y) > \theta$$

We can now write an algorithm that performs step 1 and 2 of the 3-step Privacy through Imputation process described at the start of this chapter, and we call it MII-SUPPRESS:

Algorithm 1 Mines MIIs in D using maximum support value θ. Outputs $\tilde{D}$ which is a copy of D with the MIIs removed.

```
1: procedure MII-SUPPRESS(D, θ)
2:     X ← MINEMIIS(D, θ)
3:     D̃ ← []                                  ▷ Create empty dataset
4:     for t ∈ D do
5:         X' ← {x ∈ X | x ⊆ t}                ▷ Calculate which MIIs are in t
6:         t̃ ← {i ∈ t | ∀x ∈ X', i ∉ x}        ▷ Remove MIIs from t
7:         D̃.append(t̃)
8:     return D̃
```

It is easy to see that MII-SUPPRESS gives us the following privacy guarantee:

Lemma 2.1. *Let* $\tilde{D} =$ MII-SUPPRESS(D, θ)*, then* $\forall \tilde{t} \in \tilde{D} : sup_D(\tilde{t}) > \theta$*.*

Proof. This is a simple proof by contradiction. Assume

$$\exists \tilde{t} \in \tilde{D} : sup_d(\tilde{t}) \leq \theta$$

which means that $\tilde{t}$ was an infrequent itemset in D. Every infrequent itemset is a superset of an MII because of the upward closure property. However, $\tilde{t}$ was constructed by copying t and removing all MIIs from it, so $\tilde{t}$ cannot be a superset of an MII. □

This tells us that $\tilde{D}$ only reveals itemsets that were frequent in the original dataset. However, on second thought, we can derive a stronger privacy guarantee:

Lemma 2.2. *Let* $\tilde{D} =$ MII-SUPPRESS(D, θ)*, let* $t \in D$ *be a transaction and let* X' *and* $\tilde{t}$ *be defined as in line 5 and 6 of the* MII-SUPPRESS *algorithm. Then* $\forall x \in X', y \subset x : sup_D(\tilde{t} \cup y) > \theta$

Proof. Assume the opposite, i.e. $sup_D(\tilde{t} \cup y) \leq \theta$. Then $\exists z \subseteq (\tilde{t} \cup y)$ such that z is an MII because of upward closure. We know that $z \not\subseteq y$ because $y \subset x$ and x is an MII, so $sup_D(y) > \theta$. lemma 2.1 tells us that $\tilde{t}$ was frequent in D, and we assumed z to be an MII in D, so it should be the case that $z \cap \tilde{t} = \emptyset$. However

$$\big(z \subseteq (\tilde{t} \cup y)\big) \wedge (z \not\subseteq y) \Rightarrow \exists i \in z : i \in \tilde{t}$$

which implies $z \cap \tilde{t} \neq \emptyset$. This is a contradiction. □

Although perhaps not immediately obvious, this lemma allows us to strengthen the bound in lemma 2.1.

Theorem 2.3. *Let* $\tilde{D} =$ MII-SUPPRESS(D, θ)*, let* $t \in D$ *be a transaction and let* X' *and* $\tilde{t}$ *be defined as in line 5 and 6 of the* MII-SUPPRESS *algorithm. Let* $c = \underset{x \in X'}{\operatorname{argmax}} (|x|)$*, then* $sup_D(\tilde{t}) \geq \theta + |c|$

Proof. If $|c| = 1$ then $sup_D(\tilde{t}) \geq \theta + |c|$ is identical to $sup_D(\tilde{t}) > \theta$ which was already proved in lemma 2.1. Therefore we only have to prove the case where $|c| > 1$. Let $Y = \binom{c}{|c|-1}$, i.e. Y contains all subsets of c of cardinality $|c| - 1$. We then know these two boolean expressions should simultaneously hold:

$$sup_D(\tilde{t} \cup c) \leq \theta \qquad \text{(superset of MII } c \text{ must be infrequent)} \tag{2.1}$$

$$\forall y \subset Y : sup_D(\tilde{t} \cup y) > \theta \qquad \text{(lemma 2.2)} \tag{2.2}$$

The best way to satisfy both expressions is as follows. First, let $sup_D(\tilde{t} \cup c) = \theta$, i.e. there are exactly θ transactions in D that contain both $\tilde{t}$ and c. This satisfies equation (2.1) but

not equation (2.2) because we now have $\forall y \subset Y : sup_D(\tilde{t} \cup y) = \theta$. Therefore, for every $y \subset Y$, D should contain exactly one additional transaction t such that $\tilde{t} \subseteq t, y \subseteq t, c \not\subseteq t$. This way, we satisfy both equation (2.1) and equation (2.2) because now

$$sup_D(\tilde{t} \cup c) = \theta$$
$$\forall y \subset Y : sup_D(\tilde{t} \cup y) = \theta + 1$$

There are now $\theta + |Y|$ transactions that are a superset of $\tilde{t}$. $|Y|$ contains every subset of c of size $|c| - 1$, so $|Y| = |c|$, which means that $sup_D(\tilde{t}) \geq \theta + |c|$. □

In practice, this theorem tells us that if we have a transaction t, and t contains one single MII x of size 5, then $\tilde{t} = t \setminus x$ must have $sup_D(\tilde{t}) = \theta + 5$.

We have now proved that any row $\tilde{t} \in \tilde{D}$ had a support in D that was at least $\theta + 1$, but we have also shown that the more $\tilde{t}$ was suppressed, the *higher* $\tilde{t}$'s support was in D. This is a nice consequence of the MII-SUPPRESS algorithm: the larger the MIIs in an original transaction, the stronger is the privacy guarantee MII-SUPPRESS imposes on it.

2.3 MII-suppress as a k-anonymity algorithm

We have shown in the previous section that every row in $\tilde{D}$ had a support of at least $\theta + 1$ in the original dataset. Furthermore, we have shown that the larger the MIIs in an original row, the higher the support of the sanitized row was in the original dataset. This makes it probable that MII-SUPPRESS can actually serve as a pretty good k-anonymity algorithm for $k = \theta + 1$. Such an algorithm would look as follows:

Algorithm 2 Generates sanitized dataset $\tilde{D}$ based on D by using MII-SUPPRESS, such that $\tilde{D}$ offers k-anonymity for $k = \theta + 1$.

```
1: procedure K-MII(D, θ)
2:     D̃ ← MII-SUPPRESS(D, θ)
3:     for t̃ ∈ D̃ do
4:         if sup_D̃(t̃) ≤ θ then
5:             Remove t̃ from D̃
6:     return D̃
```

All this algorithm does is that it generates $\tilde{D}$ using MII-SUPPRESS, after which it removes infrequent rows from $\tilde{D}$. K-MII clearly does not guarantee optimal k-anonymity, but we expect it to remove relatively few rows in most cases because of theorem 2.3. The interesting thing about K-MII is that it achieves k-anonymity without explicit partitioning or clustering, contrary to other well known algorithms like Mondrian[25]. Another reason why K-MII is an interesting k-anonymity algorithm, is that it only performs suppression. Many of the attacks on k-anonymity, like the background knowledge attack, are less feasible because of this. In chapter 4 we will show through experiments that K-MII removes few rows for different real-world datasets and θ values.

Although not discussed so far, MII mining is a very expensive process. Therefore, K-MII could potentially benefit from sampling: rather than mining all MIIs in D, MIIs with a lower support threshold could be drawn from a sample in D. These MIIs can then be suppressed in all of D. Mining on a sample has proved to be very effective for other mining tasks like FI mining, e.g. [38]. Depending on the type of dataset, it is possible that the suppression works very well and that very few rows are removed by K-MII due to theorem 2.3, even though the MIIs were mined from a sample of the dataset. However, this is outside the scope of this work.

2.4 Towards Differential Privacy

MII-SUPPRESS offers nice privacy guarantees as shown in the previous sections, and K-MII offers k-anonymity. Mechanisms like k-anonymity are still very much in use today. For example, Google recently tried to use k-anonymity in their Federated Learning of Cohorts[1] system, which was meant to replace third party browser cookies. More generally, k-anonymity may still be used to sanitize datasets where it is reasonable to put utility first. However, the state of the art in PPDP is differential privacy, and neither MII-SUPPRESS nor K-MII are differentially private for the very simple reason that differential privacy requires, by definition, a stochastic mechanism.

The obvious way to resolve this is to implement a stochastic imputation step. After all, although MII-SUPPRESS and K-MII solve the first two steps of the Privacy through Imputation

[1] `https://web.archive.org/web/20240119052452/https://web.dev/articles/floc`

framework, we have not actually performed the imputation step yet. We have researched this possibility, but implementing stochastic imputation on $\tilde{D}$ in such a way that we achieve a differentially private guarantee has proven to be very tricky. The difficulty lies in the fact that by imputing missing values, we implicitly create dependencies between rows with missing values and rows that we sample from during imputation. Differential privacy guarantees the privacy of individuals, i.e. individual rows, but we were unable to derive such a guarantee because individuals start to depend on each other because of stochastic sampling.

This means that we were unsuccessful in implementing a differentially private algorithm that implements Privacy through Imputation, but that does not mean that it is impossible to derive a differentially private algorithm based on the algorithms we have developed so far. Below, we will develop an algorithm, which we call Differentially Private Minimal Infrequent Itemset Suppression (DP-MII-Suppress), that works by suppressing MIIs and offers differential privacy. The algorithm uses a significantly easier stochastic mechanism: it draws a random sample with replacement from the dataset.

2.4.1 Differential Privacy under Sampling

We can develop an MII-based suppression algorithm that offers differential privacy, by using a result from Li (who also happens to be one of the authors of t-closeness[26]) et al.[27]. The algorithm from [27] is called $(\beta, \epsilon, \delta)$-DPS and works as follows. First, let $\mathcal{G}$ be a generalization scheme, i.e. a function that generalizes values values in D. $\mathcal{G}$ must be *data-independent*, meaning that the generalization rules applied to D have not been learned from D. Then, we can compute $D' = \mathcal{G}(D)$. Next, remove any row t from D' for which $sup_{D'}(t) < k$. The authors call this *k-suppression*. At this point, D' is k-anonymous. Then, calculate $\tilde{D}$ from D' by randomly sampling rows with replacement from D', with sampling rate β where $0 < \beta < 1$, and remove any row from the sample that appears $< k$ times (another round of k-suppression). The resulting dataset, $\tilde{D}$, is clearly k-anonymous, but is also differentially private. More specifically, let $f(j, n, \beta)$ be the PMF for the binomial distribution (j successes in n trials with success probability β), i.e.

$$f(j, n, \beta) = \binom{n}{j} \beta^j (1 - \beta)^{n-j}$$

and furthermore, let

$$d(k, \beta, \epsilon) = \max_{n : n \geq \left\lceil \frac{k}{\gamma} - 1 \right\rceil} \sum_{j > \gamma n}^{n} f(j, n, b)$$

where

$$\gamma = \frac{e^{\epsilon} - 1 + \beta}{e^{\epsilon}}$$

then the following can be said:

Theorem 2.4. *Li et al.[27]: $(\beta, \epsilon, \delta)$-DPS satisfies differential privacy for $\epsilon \geq -\ln(1 - \beta)$ and $\delta = d(k, \beta, \epsilon)$.*

$(\beta, \epsilon, \delta)$-DPS is mainly presented as a theoretical result by the authors, because it is the first result that proves that there is a link between k-anonymity and differential privacy. However, it is difficult to apply this algorithm in practice, because it is difficult to come up with a generalization scheme $\mathcal{G}$ that

1. is data indepentent, but at the same time
2. generalizes D in such a way that not many rows are removed during the two *k-suppression* steps, sampling, and then the second k-suppression step.

For example, one very simple data independent generalizer is $\mathcal{G}(x) = x$, i.e. the identity function, but in this case a lot of rows will likely be removed after this (faux) generalization step during the *k-suppression* step. On the other hand, we could let $\mathcal{G}$ generalize very aggressively, in which case *k-suppression* will remove relatively few rows, but the utility will be very low in that case. Most optimal would be a generalizer that strikes a balance between these two extremes, but such a balance is difficult to attain because $\mathcal{G}$ needs to be data independent.

Our attempt to solve this works as follows. First, like we did with MII-SUPPRESS, we will suppress rather than generalize. Although the authors only discuss generalization algorithms, we can apply their theorem just as well to MII-SUPPRESS because the information loss incurred through suppression is always at least equally high as any form of generalization, and therefore the privacy guarantee is at least equally strong. So, rather than developing a generalization scheme $\mathcal{G}$, we use MII-SUPPRESS as a suppression scheme. However, MII-SUPPRESS mines MIIs from D and then suppresses these MIIs in D, which means that $\mathcal{G}$ would not be data independent.

We therefore randomly partition D into two disjoint datasets D_1 and D_2 with partition rate r, i.e. $\forall r \in D : P(t \in D_1) = r$, $P(t \in D_2) = 1 - r$. We can train $\mathcal{G}$ on D_1, i.e. we mine MIIs from D_1, and then apply $\mathcal{G}$ to D_2, i.e. suppress the mined MIIs on D_2.

This calls for two different support thresholds θ_1 and θ_2, where the former is used to mine MIIs on D_1 and the latter is used to suppress MIIs in D_2. Clearly, this solution is only data independent under the assumption that the rows in D are independent from each other, but this is not an uncommon assumption for PPDP algorithms.

We call the resulting algorithm DP-MII-SUPPRESS and its pseudocode can be found below.

Algorithm 3 Mines MIIs in D using maximum support value θ. Outputs $\tilde{D}$ which is a copy of D with the MIIs removed.

1: **procedure** DP-MII-SUPPRESS(D, θ_1, θ_2, r, β)
2: $D_1, D_2 \leftarrow$ RANDOMPARTITION(D, r)
3: $X \leftarrow$ MINEMIIS(D_1, θ_1) ▷ Mine all MIIs in D_1
4: $D' \leftarrow []$
5: **for** $t \in D_2$ **do**
6: $X' \leftarrow \{x \in X \mid x \subseteq t\}$ ▷ Calculate which MIIs are in t
7: $t' \leftarrow \{i \in t \mid \forall x \in X', i \notin x\}$ ▷ Remove MIIs from t'
8: $D'.append(t')$
9: **for** $\tilde{t} \in D'$ **do**
10: **if** $sup_{D'}(\tilde{t}) \leq \theta_2$ **then** ▷ k-suppression for $k = \theta_2 + 1$
11: Remove $\tilde{t}$ from $\tilde{D}$
12: $\tilde{D} \leftarrow []$
13: **for** $i \in \{0, 1, \ldots, \lfloor \beta |D'| \rfloor\}$ **do**
14: $t \sim D'$ ▷ Sample with replacement from D'
15: $\tilde{D}.append(t)$
16: **for** $\tilde{t} \in \tilde{D}$ **do**
17: **if** $sup_{\tilde{D}}(\tilde{t}) \leq \theta_2$ **then** ▷ k-suppression for $k = \theta_2 + 1$
18: Remove $\tilde{t}$ from $\tilde{D}$
19: **return** $\tilde{D}$

The generalization scheme used in DP-MII-SUPPRESS is trained on D_1 and applied to

D_2. The final output dataset $\tilde{D}$ is a sample drawn from D_2, which makes the generalization scheme data independent. Therefore, DP-MII-SUPPRESS is a practical implementation of $(\beta, \epsilon, \delta)$-DPS that offers differential privacy for any $\epsilon \geq -\ln(1-\beta)$ and $\delta = d(\theta_2 + 1, \beta, \epsilon)$. Because of the partitioning and sampling, we know that the following upper bound holds on the height of the sanitized dataset:

$$|\tilde{D}| \leq (r-1)\beta|D| \tag{2.3}$$

but depending on parameter choices and the dataset, it is possible that $|\tilde{D}|$ ends up being smaller than this upper bound due to the two rounds of k-suppression.

Because DP-MII-SUPPRESS suppresses and $(\beta, \epsilon, \delta)$-DPS generalizes, DP-MII-SUPPRESS returns an incomplete dataset whereas $(\beta, \epsilon, \delta)$-DPS returns a complete dataset. Therefore, if we want to make the dataset appear complete, an imputation step can be performed on $\tilde{D}$ = DP-MII-SUPPRESS. It is safe to do so without affecting the privacy guarantee, because differential privacy offers post-processing immunity[17]. This makes the imputation step differentially private as well, but it does not add any additional privacy while it does further obscure to a data analyst which parts of the dataset are 'original' and which parts were imputed.

This means that DP-MII-SUPPRESS is only to a certain extent an implementation of Privacy through Imputation. Step 1 and 2 are implemented as planned: we learn which parts are too revealing by mining MIIs, and then we suppress these MIIs. However, we then perform an additional sampling step in combination with k-anonymity to achieve differential privacy. This means that the differential privacy guarantee does not come from an imputation step, but rather from sampling. In fact, DP-MII-SUPPRESS performs no imputation at all. This step can be performed after executing DP-MII-SUPPRESS if it is desirable.

2.4.2 Parameter choices

DP-MII-SUPPRESS(D, θ_1, θ_2, r, β) offers (ϵ, δ)-differential privacy for any $\epsilon \geq -\ln(1-\beta)$ and $\delta = d(\theta_2 + 1, \beta, \epsilon)$. Because some parameters depend on others, parameters should be picked in a certain order. It is common to put privacy first, i.e. to first pick a desired value for ϵ and δ, and then worry about utility. In the case of DP-MII-SUPPRESS, this means that parameters can be picked as follows:

1. Choose ϵ and δ.

2. Choose θ_2.

3. Choose β such that $\beta \leq 1 - e^{-\epsilon}$ (because we need to satisfy $\epsilon \geq -\ln(1-\beta)$) and such that $\delta = d(\theta_2 + 1, \beta, \epsilon)$. d is not a closed form expression, so we cannot calculate β directly. However, we can easily binary search the value β in the interval $(0, 1 - e^{-\epsilon}]$ in order to satisfy $\delta = d(\theta_2 + 1, \beta, \epsilon)$. It is possible that even when β is maximized, i.e. $\beta = 1 - e^{-\epsilon}$, we get $d(\theta_2 + 1, \beta, \epsilon) < \delta$. In other words: even when the curator picks the largest sample rate that ϵ allows, they still end up with a δ that is smaller than the δ they required. In this case, the curator ends up with 'free' additional privacy.

4. Once $\epsilon, \delta, \theta_2, \beta$ have all been chosen, θ_1 and r remain. These parameters are in no way related to privacy and only impact utility. If the dataset has a clear utility target, then a grid search with θ_1 and r can be performed in order to find optimal values. However, in many cases this will not be realistic. In such cases, it could suffice to experiment with a few values for r and pick $\theta_1 = \left\lceil \frac{(\theta_2+1)r}{\beta(1-r)} \right\rceil$. The reason for this is that DP-MII-SUPPRESS performs k-suppression on D_2 with $k = \theta_2 + 1$, it then draws a sample with rate β, and then performs k-suppression again. Given that rows need to be at least k-frequent on the sample of size $\leq \beta|D_2|$, they should approximately be $\geq \frac{k}{\beta}$-frequent on D_2 prior to sampling. The equivalent support threshold θ_1 for D_1 should therefore be at least

$$\frac{\frac{k}{\beta}}{|D_2|} \cdot |D_1| = \frac{k|D_1|}{\beta|D_2|} = \frac{(\theta_2 + 1)r}{\beta(1-r)}$$

and because θ_1 needs to be discrete, this expression is ceiled.

2.4.3 Utility

Like many privacy preservation algorithms, DP-MII-SUPPRESS puts privacy first and allows us to quantify the amount of privacy loss caused by publishing a sanitized release because of the differentially private guarantee. This means that DP-MII-SUPPRESS does not offer an explicit utility guarantee, although we will measure utility experimentally for various parameter choices in chapter 4.

One thing that will generally impact utility is the number of MIIs that are mined. More MIIs roughly equates to more suppression and lower utility. We can experimentally analyze

how many MIIs certain datasets contain in practice, but from a theoretic standpoint it is also interesting to look at if there is a tight upper bound on the number of MIIs in a rectangular categorical dataset. We will research this topic in the next chapter.

Chapter 3

Complexity analysis of MII mining on categorical datasets

In the previous chapter, we have developed algorithms that use MIIs in order to sanitize categorical datasets. One question that naturally arrises is: how many MIIs can a categorical dataset contain, and how fast can we find them? We will discuss these questions in this chapter. In the next section we will first briefly relate MII mining to the most well-known itemset mining task there is: frequent itemset (FI) mining. We will briefly discuss the complexity of FI mining, after which we will offer a detailed analysis of the complexity of MII mining. Among other things, we will prove that the upper bound on the number of MIIs in an $m \times n$ dataset is $m\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Theta\left(m2^{n-\frac{\log(n)}{2}}\right)$ for any dataset width n and any support threshold θ, and we will prove that this upper bound is tight.

3.1 Complexity of FI mining

In itemset mining, every transaction in a dataset D is a subset of $\mathcal{I}$, which is the set of all items present in D. In frequent itemset (FI) mining, every itemset X in dataset D is mined for which $sup_D(X) \geq \theta$. The family of frequent itemsets, which we will denote by $\mathcal{F}_{FI}$, is closed under set inclusion, i.e. $\forall J \subseteq I : I \in \mathcal{F}_{FI} \Rightarrow J \in \mathcal{F}_{FI}$. Because of this downward closure property it is very trivial to construct a dataset with the maximum possible number of frequent itemsets for a given $\mathcal{I}$ and θ: simply repeat $\mathcal{I}$ θ times. Because $\mathcal{I}$ appears θ times,

it is a frequent itemset and because of downward closure, so is every subset. This means that D contains $2^{|\mathcal{I}|} - 1$ frequent itemsets, which is maximal. This means that the output of an FI mining algorithm can be exponential with respect to $|\mathcal{I}|$, which automatically proves that the worst case time complexity must also be at least exponential with respect to $|\mathcal{I}|$, because outputting an exponential number of itemsets requires exponential time.

However, in the real world, transactions will be significantly less wide than $|\mathcal{I}|$. More specifically, in the context of PPDP the datasets are often tables of a fixed width n, so we focus on rectangular $m \times n$ datasets. If we use such a dataset design to analyze the complexity of FI mining, we can still show relatively easily that in the worst case, FI mining must take at least exponential time and space with respect to n. A proof sketch:

Proof. Let $m, n, \theta \in \mathbb{N}$, with $\theta \leq m$. Construct a $\lfloor \frac{m}{\theta} \rfloor \times n$ matrix D such that all items in D are unique. For every row in D, add $\theta - 1$ exact copies to D. D now has height $\theta \lfloor \frac{m}{\theta} \rfloor \approx m$. For every row $r \in D$, we know that every itemset $X \in (\mathcal{P}(r) - \{\emptyset\})$ has $sup_D(X) = \theta$. In other words: all subsets of r are frequent. $|\mathcal{P}(r) - \{\emptyset\}| = 2^n - 1$, and we have $\lfloor \frac{m}{\theta} \rfloor$ unique rows in D. This gives $(2^n - 1)\lfloor \frac{m}{\theta} \rfloor$ frequent itemsets. The average size of a member of $(\mathcal{P}(r) - \{\emptyset\})$ is equal to $\frac{n2^{n-1}}{2^n - 1}$, which is on the order of $\frac{n}{2}$ and thus $\mathcal{O}(n)$. This means the asymptotic output space complexity of FI mining is $(2^n - 1)\lfloor \frac{m}{\theta} \rfloor \mathcal{O}(n) = \mathcal{O}(\frac{nm}{\theta} 2^n)$. This shows that in the worst case, mining all FI's must take at least exponential space and thus exponential time with respect to n. □

Again, this proof was very simple because thanks to the downward closure property we only had to construct a matrix with a maximum number of frequent *rows* in order to yield the matrix with a maximum number of frequent *itemsets*.

3.2 Frequent and infrequent mining algorithms

$\mathcal{F}_{FI}$ is closed under set inclusion. Sadly, the set of all MIIs, denoted by $\mathcal{F}_{MII}$, is by definition not closed under set inclusion, because an itemset can only be an MII if all its proper subsets are *not* MIIs. Although $\mathcal{F}_{FI}$ has downward closure and $\mathcal{F}_{MII}$ has upward closure, there are still striking similarities between FI and MII mining. For example, Apriori[2] is a well known FI mining algorithm that makes use of the downward closure property, but it implicitly mines

all MIIs as well.

Algorithm 4 Given θ and an $m \times n$ dataset D, returns $\mathcal{F}_{FI}$, i.e. the family of all itemsets that appear more than θ times in D.

1: **procedure** APRIORI(m, n, D, θ)
2: $k \leftarrow 1$, $C^1 = \{i \mid i \text{ is a 1-itemset in D}\}$
3: **while** $C_k \neq \emptyset$ **do**
4: $\mathcal{F}_{FI}^k \leftarrow \{c \mid c \in C^k, sup(c, D) > \theta\}$
5: $C^{k+1} \leftarrow \{c \mid c_1, c_2 \in \mathcal{F}_{FI}^k, c = c_1 \cup c_2, |c| = k + 1\}$
6: $C^{k+1} = \{c \mid c \in C^{k+1}, \forall d \in c^{[k]} : d \in \mathcal{F}_{FI}^k\}$ ▷ Prune c's with infrequent k-subset
7: $k \leftarrow k + 1$
8: **return** $\bigcup_{j=1}^{k} \mathcal{F}_{FI}^j$

Apriori generates k-candidates, i.e. k-itemsets whose proper subsets are all frequent. Then, on line 4, it calculates $\mathcal{F}_{FI}^k$ by throwing away all infrequent k-candidates. However, the candidates that are thrown away in this step are precisely all MIIs of size k. Because of this, Apriori can trivially be converted into an MII-mining algorithm, which also tells us that MII can be done in the same time complexity that Apriori has.

Different FI mining algorithms that do not use candidate generation, like FP-Growth[20], cannot be converted to MII mining algorithms quite as easily, but the conversion often *is* still possible. As such, FP-Growth has been rewritten into IFP-Growth in 2012[18].

3.2.1 Comparison of complexities of FI and MII mining

As we have shown, mining all FI's takes exponential time and space with respect to n. Furthermore, it is not hard to see that the problem of finding a single FI in D, if one exists, is easier: iterate over all mn items in the dataset (there are possibly duplicates among them), count their supports using a hashset (or a similar data structure that accommodates constant amortised lookup, insertion and updating), and return the first item whose suport reaches the frequency threshold.

Mining a single MII from D is more challenging however: an infrequent singleton itemset can be found in $O(mn)$ time by traversing and counting all items, but if all singletons are frequent then we must look at all 2-sets. If all 2-sets are frequent, we must search for through

the 3-sets, and so on. On the other hand, when we want to mine a single FI we never have to consider more than the singleton itemsets because $\mathcal{F}_{FI}$ is closed under set inclusion.

So, mining a single MII is more challenging than mining a single FI. But what about mining *all* MIIs? Can we derive a tight bound on the maximum number of MIIs? And can we do this for categorical datasets, in which certain items cannot co-occur in the same transaction because they belong to the same category? Such a tight bound is mainly interesting from a theoretical standpoint, but is also interesting for people who are actually mining MIIs to have a ballpark estimate of how many MIIs can be found in the absolute worst case in their dataset.

In the remainder of this chapter we will show that, similar to frequent itemset mining, the number of MIIs in a dataset is exponential with respect to the dataset width n, but that the bound is a bit tighter than the bound for FI mining. More specifically, we will prove the following statements:

1. Any $m \times n$ matrix contains at most $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs.
2. For every n and θ, matrices exist that contain precisely the maximum number of MIIs, which makes the space complexity of mining all MIIs exponential with respect to the dataset width n.
3. For every n and θ there are in fact infinitely many matrices, each of distinct prime power height, that contain the maximum number of MIIs that their height permits.
4. For every n and $\theta \geq 2$, if m is sufficiently large (but not neccessarily a prime power), we can always construct an $m \times n$ matrix that contains $\Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right)$ MIIs.

3.3 A tight bound on the number of MIIs in a categorical dataset

First, recall the two requirements an itemset must satisfy in order to be an MII:

1. the itemset itself can appear at most θ times in D
2. all subsets of the itemset must appear *more than* θ times in D.

Also take note of the following observations:

Observation 1. *No MII can appear twice in a row.*

This observation always holds, because the dataset contains categorical values, and every categorical value can only appear in one category, i.e. one column.

Observation 2. *For any pair of two distinct MIIs a, b:* $a \not\subset b \wedge b \not\subset a$.

This is implied by the definition of an MII, because if $a \subset b$, then either b contains a subset that is infrequent, or a itself is frequent. In both cases one of the itemsets is not an MII. Do note however, that this observation does not imply that two MIIs cannot *partially* overlap, because they definitively can.

Now, we can first look at the maximum number of MIIs that a single row in dataset D can contain.

Lemma 3.1. ***Maximum number of MIIs per row.*** *A single row of width n cannot contain more than* $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ *MIIs.*

Proof. Let S be a set with n members and let $\mathcal{F}$ be a family of subsets of S, such that for every distinct pair of sets $a, b \in \mathcal{F} : a \not\subseteq b \wedge b \not\subseteq a$. Sperner's Theorem tells us that $|\mathcal{F}|$ is maximized when $\mathcal{F} = \left\{ s \subseteq S \;\middle|\; |s| = \lfloor \frac{n}{2} \rfloor \right\}$, i.e. when $\mathcal{F}$ contains every subset of S of size $\lfloor \frac{n}{2} \rfloor$. We know that MIIs can never contain other MIIs (see observation 2), and therefore we also know that a row of width n cannot contain more than $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs. □

$\mathcal{F}_{MII}$ represents the family of all MIIs in an $m \times n$ categorical dataset D. The following can be said about the maximum size of $\mathcal{F}_{MII}$:

Theorem 3.2. ***Maximum MIIs in a dataset.*** $|\mathcal{F}_{MII}| \leq m\binom{n}{\lfloor \frac{n}{2} \rfloor}$

Proof. lemma 3.1 shows that one row can contain at most $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs. Therefore, a complete dataset of m rows can never contain more than $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs. □

This bound is exponential with respect to n, which becomes more apparent by using Stirling's approximation for the factorial function:

$$n! \sim \sqrt{2n\pi}\left(\frac{n}{e}\right)^n \qquad \text{(Stirling's approximation)}$$

When n is even, this yields

$$\begin{aligned} m\binom{n}{\lfloor \frac{n}{2} \rfloor} = m\binom{n}{\frac{n}{2}} &= m\frac{n!}{\frac{n}{2}! \cdot \frac{n}{2}!} \\ &\sim m\frac{\sqrt{2n\pi}\left(\frac{n}{e}\right)^n}{\left(\sqrt{n\pi}\left(\frac{n}{2e}\right)^{\frac{n}{2}}\right)^2} \\ &\sim m\frac{\sqrt{2n\pi}\ n^n\ e^{-n}}{n\pi\ n^n\ 2^{-n}\ e^{-n}} \\ &\sim m\frac{2^n}{\sqrt{\frac{n\pi}{2}}} \end{aligned}$$

We can get rid of the constants in the denominator, but at that point the left and right hand side of the equation are no longer asymptotically equal, so we have to start using Θ-notation:

$$= \Theta\left(m\frac{2^n}{\sqrt{n}}\right)$$

And at this point we can get rid of the fraction completely:

$$= \Theta\left(m2^{n-\frac{\log(n)}{2}}\right)$$

This approximation is only valid when n is even, but is still an easy way to demonstrate the exponential nature of $|\mathcal{F}_{MII}|$.

By now we have an upper bound on $|\mathcal{F}_{MII}|$. In the next section, we will show that this upper bound is tight for all n and θ.

3.3.1 Proof of tightness using Orthogonal Arrays

We will prove that the upper bound in theorem 3.2, $|\mathcal{F}_{MII}| \leq m\binom{n}{\lfloor \frac{n}{2} \rfloor}$, is tight for all n and θ by showing that for all n and θ matrices exist that contain precisely this number of MIIs. We do this by using the concept of *orthogonal arrays*.

Definition 3.3.1. Orthogonal Array (OA) Let S be a set with s members, let λ, t, n be positive integers, and let $n \geq t$. A 2D array of dimensions $\lambda s^t \times n$, where every cell in the array contains a value that is a member of S, is called an orthogonal array, denoted by $OA_\lambda(n, s, t)$, iff in every $\lambda s^t \times t$ submatrix, every possible t-tuple that can be constructed from members of S appears exactly λ times.

An example of an orthogonal array is given in table 3.1. The height of an orthogonal array is λs^t because one can construct s^t unique t-tuples from the s-sized set S, and every t-tuple has to appear exactly λ times in every $\lambda s^t \times t$ submatrix.

Note that an orthogonal array differs slightly from how we have defined datasets so far; in our case, an item always appears in a single column, whereas in the case of orthogonal arrays every item appears in *every* column. However, essentially they are both the same thing: we can just think of the numbers 1, 2, 3, ... in any column j as the first, second, third, ... value of the j'th categorical variable. This also means that the itemset $\{1, 2\}$ in column pair 3, 4 is *not* the same as itemset $\{1, 2\}$ in column pair 5, 9. After all, item 1 in column 3 represents a different value than item 1 in column 5, and the same is true for item 2 in column 4 or 9.

$$\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 2 & 2 & 2 & 2 \\
1 & 2 & 2 & 1 & 1 & 2 & 2 \\
1 & 2 & 2 & 2 & 2 & 1 & 1 \\
2 & 1 & 2 & 1 & 2 & 1 & 2 \\
2 & 1 & 2 & 2 & 1 & 2 & 1 \\
2 & 2 & 1 & 1 & 2 & 2 & 1 \\
2 & 2 & 1 & 2 & 1 & 1 & 2
\end{bmatrix}$$

Table 3.1: $\mathrm{OA}_2(7,2,2)$ with $S = \{1, 2\}$. As is in accordance with the parameters of the orthogonal array, the dimensions are $2 \cdot 2^2 \times 7$, and each of the four 2-tuples in $S \times S$, i.e. $(1, 1), (1, 2), (2, 1), (2, 2)$, appears exactly twice in every 8×2 submatrix. For instance, take the tuple $(1, 2)$, pick any pair of two columns, and notice how $(1, 2)$ will appear exactly twice in the column pair you picked. For more concrete examples, we have colored the two occurrences of $(1, 2)$ red in column pair 3 and 7, and blue in column pair 4 and 6.

We can now prove the tightness of the bound in theorem 3.2 by using a few lemmas.

Lemma 3.3. *In any $OA_\lambda(n,s,t)$, every $(t-1)$-tuple appears precisely λs times in every submatrix of width $t-1$.*

Proof. Every t-tuple has to appear exactly λ times in every collection of t columns of the

array. There are s^t unique t-tuples that all have to satisfy this constraint. Since every $(t-1)$-subtuple appears equally often in this set of t-tuples, and since there can only exist s^{t-1} unique $(t-1)$-tuples, every $(t-1)$-tuple must appear $\frac{\lambda s^t}{s^{t-1}} = \lambda s$ times in every $\lambda s^t \times (t-1)$ submatrix. □

From now on, we will exclusively work with orthogonal arrays with $\lambda = 1$. For such arrays, we can use lemma 3.3 to get to the following result:

Lemma 3.4. *Any $OA_1(n,s,\lfloor \frac{n}{2} \rfloor)$, with $s > \theta$, contains the maximum number of MIIs.*

Proof. Every tuple x of size $\lfloor \frac{n}{2} \rfloor$ appears exactly once in every $\lambda s^{\lfloor \frac{n}{2} \rfloor} \times \lfloor \frac{n}{2} \rfloor$ submatrix, because $\lambda = 1$. Every tuple of size $\lfloor \frac{n}{2} \rfloor - 1$ appears exactly $\lambda s = s$ times in the array, as shown in lemma 3.3. Logically, this implies that even smaller tuples must also all have a support $\geq s$. This makes every $\lfloor \frac{n}{2} \rfloor$-tuple x an MII, because:

1. x itself is infrequent, as it has a support of 1, and $1 \leq \theta$ for any valid θ (because θ must always be a positive integer);

2. all proper subsets of x appear at least $\lambda s = s$ times in the array, and since $s > \theta$ all proper subsets of x are frequent.

So: every $\lfloor \frac{n}{2} \rfloor$-tuple in the orthogonal array is an MII. There are $\lambda s^{\lfloor \frac{n}{2} \rfloor} = s^{\lfloor \frac{n}{2} \rfloor}$ rows, and $\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs per row, which gives a total of $s^{\lfloor \frac{n}{2} \rfloor}\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs in an array of dimensions $s^{\lfloor \frac{n}{2} \rfloor} \times n$. This number is exactly equal to the upper bound shown in theorem 3.2. □

This lemma shows us that any $\mathrm{OA}_1(\mathrm{n,s,}\lfloor \frac{n}{2} \rfloor)$ with $s > \theta$, contains exactly the number of MIIs the upper bound in theorem 3.2 describes. Thus, in order to show that the bound in theorem 3.2 is tight for all n, θ, we have to show that $\mathrm{OA}_1(\mathrm{n,s,}\lfloor \frac{n}{2} \rfloor)$ must exist for all n, θ. This brings is to the following theorem, proved by K.A. Bush[8]:

Theorem 3.5. *K.A. Bush: Let $s = p^x$, where p is some prime and x is a positive integer, then $\forall t < s : \exists\ OA_1(s+1,s,t)$.*

Bush actually shows how one can *construct* such orthogonal arrays using polynomials with coefficients that range over a Galois field (hence the requirement that s is a prime power). An example of the kind of orthogonal arrays Bush researched is $\mathrm{OA}_1(5+1, 5, 2)$, which can

be viewed (in transposed form) in table 3.2. This orthogonal array does not appear in Bush's paper; we generated it ourselves using Bush's method.

As we expect from $\mathrm{OA}_1(5+1,5,2)$, every 2-tuple has a support of 1 in every $\lambda s^t \times t$ submatrix, and in line with lemma 3.3, every 1-tuple has a support of $\lambda s = 5$ in every $\lambda s^t \times 1$ submatrix. Therefore, if $\theta \in \{1,2,3,4\}$, $\mathrm{OA}_1(5+1,5,2)$ contains $25\binom{6}{2} = 375$ distinct MIIs, each of size 2. When $\theta \geq 5$, the 2-tuples are still infrequent but their proper subsets, the 1-tuples, are now *also* infrequent because their support is 5 and $5 \not> \theta$. Therefore, the 2-tuples are no longer MIIs, but now the 1-tuples are MIIs. Thus, when $\theta \geq 5$, only the singleton items are MIIs. There are 5 distinct items per column and 6 columns, which yields 30 MIIs.

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 4 & 4 & 4 & 4 & 4 \\
0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 & 0 & 1 & 2 & 3 & 4 \\
0 & 2 & 4 & 1 & 3 & 4 & 1 & 3 & 0 & 2 & 3 & 0 & 2 & 4 & 1 & 2 & 4 & 1 & 3 & 0 & 1 & 3 & 0 & 2 & 4 \\
0 & 3 & 1 & 4 & 2 & 3 & 1 & 4 & 2 & 0 & 1 & 4 & 2 & 0 & 3 & 4 & 2 & 0 & 3 & 1 & 2 & 0 & 3 & 1 & 4 \\
0 & 4 & 3 & 2 & 1 & 2 & 1 & 0 & 4 & 3 & 4 & 3 & 2 & 1 & 0 & 1 & 0 & 4 & 3 & 2 & 3 & 2 & 1 & 0 & 4 \\
0 & 1 & 2 & 3 & 4 & 4 & 0 & 1 & 2 & 3 & 3 & 4 & 0 & 1 & 2 & 2 & 3 & 4 & 0 & 1 & 1 & 2 & 3 & 4 & 0
\end{bmatrix}$$

Table 3.2: $\mathrm{OA}_1(5+1,5,2)^T$. Because $s = 5^1$ is a prime power, and because $s > t$, theorem 3.5 tells us that $\mathrm{OA}_1(5{+}1,5,2)$ must exist. And indeed, it does.

Note that we can remove up to 4 columns from $\mathrm{OA}_1(5+1,5,2)$ and we will always end up with a smaller array that is still orthogonal. More generally, we can prove the following

Lemma 3.6. *Given $OA_\lambda(n,s,t)$, we can construct $OA_\lambda(x,s,t)$ $\forall x \in \{t, t{+}1, \ldots, n\}$ by taking $OA_\lambda(n,s,t)$ and removing $n-x$ columns from it.*

Proof. By definition, every $\lambda s^t \times t$ submatrix of $\mathrm{OA}_\lambda(n,s,t)$ must contain every t-tuple λ times. By removing columns, we do not introduce *new* submatrices. Rather, we end up with *fewer* submatrices, each of which already present in the original orthogonal array. Therefore the new array must still be orthogonal. □

At this point we can prove that the bound in theorem 3.2 is tight for all n, θ:

Theorem 3.7. *Let n, θ, x be positive integers, let p be any prime number, and let $s = p^x$ be*

a power of that prime number. If $s > \max(\theta, n-1)$*, then there exists a matrix of dimensions* $s^{\lfloor \frac{n}{2} \rfloor} \times n$ *that contains precisely* $s^{\lfloor \frac{n}{2} \rfloor} \binom{n}{\lfloor \frac{n}{2} \rfloor}$ *MIIs.*

Proof. We want to construct an $s^{\lfloor \frac{n}{2} \rfloor} \times n$ matrix that contains $s^{\lfloor \frac{n}{2} \rfloor} \binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs, i.e. the maximum amount. We know from theorem 3.5 by Bush that we can construct $\mathrm{OA}_1(s+1, s, t)$ as long as $s > t$. In our case, $t = \lfloor \frac{n}{2} \rfloor$. Because we required that $s > \max(\theta, n-1)$, and because $\forall n > 0 : n - 1 \geq \lfloor \frac{n}{2} \rfloor$, we know that $s > \lfloor \frac{n}{2} \rfloor$. Thus, we know that we can construct $\mathrm{OA}_1(s+1, s, \lfloor \frac{n}{2} \rfloor)$, which has dimensions $s^{\lfloor \frac{n}{2} \rfloor} \times (s+1)$. We required that $s > n-1$, so $s+1 > n$, meaning that this array is always wider than n. However, we have proved in lemma 3.6 that we can construct $\mathrm{OA}_1(n, s, \lfloor \frac{n}{2} \rfloor)$ from $\mathrm{OA}_1(s+1, s, \lfloor \frac{n}{2} \rfloor)$ as long as $s + 1 \geq n \geq \lfloor \frac{n}{2} \rfloor$, which is always true.

At this point we know that $\mathrm{OA}_1(n, s, \lfloor \frac{n}{2} \rfloor)$ must exist. We have proved in lemma 3.4 that $\mathrm{OA}_1(n, s, \lfloor \frac{n}{2} \rfloor)$ contains the maximum number of MIIs as long as $s > \theta$. We satisfy this requirement because we required that $s > \max(\theta, n-1)$. □

This theorem shows us that for any infrequency threshold θ, and any dataset width n, a matrix must exist that contains exactly the maximum number of MIIs as was specified by the bound in theorem 3.2. This proves the tightness of theorem 3.2 for all n, θ.

3.4 ∞

The previous section proved the tightness of theorem 3.2 for all n, θ by showing that for any n and θ we can construct a matrix that contains the maximum number of possible MIIs. However, there is not just one matrix for which the bound in theorem 3.2 is tight. After all, Euclid's theorem tells us that there must always be yet a larger prime, and there are infinitely many powers of that prime, so theorem 3.5 by Bush tells us that there is always yet a taller matrix we can construct that contains the maximum possible number of MIIs. This implies that for any n and θ, we can construct infinitely many matrices, each of a distinct height, that contain the maximum number of MIIs that their height permits. However, these matrix heights grow very quickly, as can be seen in table 3.3.

As an example of how to read table 3.3, take $n = 10$, $\theta = 20$. As can be seen in the table, the third smallest prime power $s > \max(\theta, n-1)$ is 27. For these parameters, we would

n	θ	Smallest prime powers s such that $s > \max(\theta, n-1)$	Heights $s^{\lfloor \frac{n}{2} \rfloor}$
5	1	5, 7, 8, 9, 11	25, 49, 64, 81, 121
5	10	11, 13, 16, 17, 19	121, 169, 256, 289, 361
10	1	11, 13, 16, 17, 19	161051, 371293, 1048576, 1419857, 2476099
10	20	23, 25, 27, 29, 31	6436343, 9765625, 14348907, 20511149, 28629151
100	1	101, 103, 107, 109, 113	$\approx 10^{100.216}$, $10^{100.642}$, $10^{101.469}$, $10^{101.871}$, $10^{102.654}$
100	200	211, 223, 227, 229, 233	$\approx 10^{116.214}$, $10^{117.415}$, $10^{117.801}$, $10^{117.992}$, $10^{118.368}$

Table 3.3: Smallest orthogonal array heights that satisfy theorem 3.7, for given n and θ.

end up with a matrix of dimensions $s^{\lfloor \frac{n}{2} \rfloor} \times n = 14,348,907 \times 10$, (note that 14348907 is indeed the third value in the rightmost column) and this is the third *smallest* matrix we can construct this way. In this matrix every $\lfloor \frac{n}{2} \rfloor$-tuple, i.e. every 5-tuple, is an MII. This yields $14,348,907\binom{10}{5} = 3,615,924,564$ MIIs and this is the maximum amount for a matrix of these dimensions.

This matrix may seem incredibly tall, but this makes a lot of sense. After all, we wanted a matrix of width $n = 10$ in which every 5-tuple appears exactly once and every 4-tuple appears more than $\theta = 20$ times. These are very difficult constraints to satisfy simultaneously, hence the tall matrix. Also, notice in the other rows in the table that, although the minimum matrix height may be huge for a certain n and θ, the successive matrix heights generally do not grow that quickly. Again, this is not particularly surprising when we look at distances between primes (Bertrand's postulate) and prime powers.

3.5 ∞ without the primes

We now know that for any θ and n we can construct infinitely many matrices, each with a distinct height, that all contain the maximum number of MIIs as long as their heights are sufficiently large prime powers. On the other hand, if m is not a prime power, it turns out to be impossible in most cases to construct an orthogonal array with the maximum number of MIIs. We concluded this ourselves after writing heavily optimized multithreaded code in Rust

(which was necessary because the search space for this problem grows spectacularly quickly once matrix dimensions increase) that searched for such matrices for weeks, but relatively few exist. One notable exception is when m is a multiple of an admissible prime power height. After all, if we have an orthogonal array of dimensions $m \times n$ with the maximum number of MIIs, then we can simply duplicate the array and map the items to a new set of unused items. We can then stitch the two arrays together vertically and end up with an orthogonal array of dimensions $2m \times n$ that contains the maximum number of MIIs. However, the resulting array will still always have a height that is a multiple of a prime power.

If we allow ourselves to be slightly more specific about which values for m and θ we permit, we can arrive at a pretty interesting result that also works for matrices whose heights are not (multiples of) prime powers:

Theorem 3.8. *Let $n > 0, \theta \geq 2, x > 0$, let p be any prime number, and let $s = p^x$ be a power of that prime number. If $s > \max(\theta, n-1)$, then for any $m \geq s^{\lfloor \frac{n}{2} \rfloor}$ there exists an $m \times n$ matrix that contains $\Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right)$ MIIs.*

Proof. Construct $j = \left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor$ distinct matrices $A_1, A_2, \ldots A_j$, each of dimensions $s^{\lfloor \frac{n}{2} \rfloor} \times n$, that all satisfy theorem 3.7, and make sure that every matrix has distinct items. Note that $m \geq s^{\lfloor \frac{n}{2} \rfloor}$, so $\left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor \geq 1$ which means that we will always construct at least one $s^{\lfloor \frac{n}{2} \rfloor} \times n$ matrix. Per theorem 3.7, each of the matrices we construct contains exactly $s^{\lfloor \frac{n}{2} \rfloor}\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs, which is the maximum amount for a $s^{\lfloor \frac{n}{2} \rfloor} \times n$ matrix as shown in theorem 3.2. By vertically stitching the matrices together, one matrix M' of width n and height $m' = s^{\lfloor \frac{n}{2} \rfloor}\left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor$ is formed. Because $A_1, A_2, \ldots, A_j$ each have unique items, we know that for every itemset x that appears in matrix A_i, $\sup(x, A_i) = \sup(x, M')$. Therefore, M' contains $m'\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs, which is the maximum possible number of MIIs for a $m' \times n$ matrix as shown in theorem 3.2. At this point, the only remaining issue is that m' is not necessarily equal to m, so there are two cases we have to cover:

1. If $s^{\lfloor \frac{n}{2} \rfloor}\left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor = s^{\lfloor \frac{n}{2} \rfloor}\frac{m}{s^{\lfloor \frac{n}{2} \rfloor}}$, then $m' = m$, which means that we have constructed an $m \times n$ matrix with $m\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right)$ MIIs, which is in line with this theorem.
2. If $s^{\lfloor \frac{n}{2} \rfloor}\left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor \neq s^{\lfloor \frac{n}{2} \rfloor}\frac{m}{s^{\lfloor \frac{n}{2} \rfloor}}$, then $m' < m$, which means that we have not actually

constructed an $m \times n$ matrix. In this case, M' lacks $m - m' = m - s^{\lfloor \frac{n}{2} \rfloor} \left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor = \left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right)$ rows. We can fix this by taking M' and "extending it" by randomly duplicating $\left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right)$ distinct rows. Note that we can always duplicate *distinct* rows, i.e. we never have to duplicate the same row multiple times, because:

(a) M' has m' rows. $m' = s^{\lfloor \frac{n}{2} \rfloor} \left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor$ and we showed at the start of this proof that $\left\lfloor \frac{m}{s^{\lfloor \frac{n}{2} \rfloor}} \right\rfloor \geq 1$, so $m' \geq s^{\lfloor \frac{n}{2} \rfloor}$.

(b) We have to duplicate $\left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right)$ rows from M'. By definition of the mod operator, $\left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right) \leq s^{\lfloor \frac{n}{2} \rfloor} - 1$. Therefore we always have to duplicate at most $s^{\lfloor \frac{n}{2} \rfloor} - 1$ rows, and we just showed that M' always has at least $s^{\lfloor \frac{n}{2} \rfloor}$ rows that we can choose from, so we never have to duplicate the same row from M' multiple times.

After the duplication step, we end up with an $m \times n$ matrix M. Because we have duplicated some rows, we know that some MIIs that had a support of 1 in M' have a support of 2 in M. However, because our theorem requires that $\theta \geq 2$, every itemset that is an MII in M' is still an MII in M. Because M' contained $m'\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs, this means that M also contains $m'\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs. In order to complete the proof of this theorem, we now have to prove that

$$m'\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right).$$

Recall that we prove $f(x) = \Omega(g(x))$ by proving that $\exists k > 0 \exists y \forall x > y : f(x) > kg(x)$. Take $k = \frac{1}{2}, y = 0$, we then have to prove

$$m'\binom{n}{\lfloor \frac{n}{2} \rfloor} > \frac{m}{2}\binom{n}{\lfloor \frac{n}{2} \rfloor}$$

Eliminate the binomial coefficient because it is > 0 for all n, and rewrite m

$$\begin{aligned} m' &> \frac{m' + \left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right)}{2} \\ \frac{m'}{2} &> \frac{m \bmod s^{\lfloor \frac{n}{2} \rfloor}}{2} \\ m' &> m \bmod s^{\lfloor \frac{n}{2} \rfloor} \end{aligned}$$

We can prove this inequality by proving that it is true when the lhs is minimized and the rhs is maximized. As shown earlier, $m' \geq s^{\lfloor \frac{n}{2} \rfloor}$ and $\left(m \bmod s^{\lfloor \frac{n}{2} \rfloor}\right) \leq s^{\lfloor \frac{n}{2} \rfloor} - 1$. Minimizing the lhs and maximizing the rhs then gives us

$$s^{\lfloor \frac{n}{2} \rfloor} > s^{\lfloor \frac{n}{2} \rfloor} - 1.$$

This inequality is true $\forall s, n$ and we have therefore proved that $m'\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right)$

□

Because this is the first proof where an MII sometimes appears twice in the matrix instead of just once, we had to change "for any θ" to "for any $\theta \geq 2$". Thanks to this slight change, we can suddenly construct matrices of non prime power (multiple) heights that contain a number of MIIs that may not be precisely $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$, but is still asymptotically bounded below by $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$.

3.6 Non-OA matrices that contain $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs

We have now completed the last proof of this chapter. We have used orthogonal arrays in our proofs because their high degree of regularity makes them easier to reason about. However, there are also many $m \times n$ matrices that contain $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs while they are *not* orthogonal arrays. In fact, for a given n and θ, the smallest matrix is often *not* an orthogonal array. For example, compare the matrices in table 3.4. The 6×4 matrix on the left contains 36 MIIs, which is the maximum amount for its dimensions. However, it is *not* an orthogonal array because the leftmost column only contains the items $\{0, 1\}$ whereas the other columns contain the items $\{0, 1, 2\}$. The 9×4 matrix on the right *is* an a valid orthogonal array, i.e. $\mathrm{OA}_1(4, 3, 2)$.

As we increase n, the difference in size grows very quickly. For instance, when $n = 6, \theta = 1$, the smallest matrix with the maximum number of MIIs has dimensions 18×6 (table A.1 in the appendices), whereas the smallest orthogonal array for these parameters is $\mathrm{OA}_1(6, 5, 3)$ with dimensions 125×6 (table A.2 in the appendices). Although the orthogonal array is almost 7 times larger than the smaller matrix, there is also a flip side: $\mathrm{OA}_1(6, 5, 3)$ *also* contains

the maximum number of MIIs when $\theta \in \{2, 3, 4\}$, whereas the number of MIIs in the 18×6 matrix immediately drops as soon as we increment θ to 2.

$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 2 & 2 & 2 \\ 1 & 0 & 2 & 1 \\ 1 & 1 & 0 & 2 \\ 1 & 2 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 2 \\ 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 2 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 2 & 1 \\ 2 & 2 & 2 & 0 \\ 2 & 1 & 0 & 1 \\ 2 & 0 & 1 & 2 \end{bmatrix}$
Smallest matrix with maximum number of MIIs ($= 6\binom{4}{2}$=36 MIIs).	Smallest orthogonal array with maximum number of MIIs ($= 9\binom{4}{2}$=54 MIIs).

Table 3.4: Comparison of smallest possible matrix and smallest orthogonal array when $n = 4$ and $\theta = 1$.

3.7 In summary

We have proved in this chapter:

1. that an $m \times n$ matrix can never contain more than $m\binom{n}{\lfloor \frac{n}{2} \rfloor}$ MIIs;
2. that for every n and θ, matrices exist that contain precisely the maximum number of MIIs, which makes the task of mining all MIIs exponential with respect to the dataset width n;
3. that for every n and θ there are in fact infinitely many matrices, each of distinct prime power height, that contain the maximum number of MIIs that their height permits;
4. that for every n and $\theta \geq 2$, if m is sufficiently large, we can always construct an $m \times n$ matrix that contains $\Omega\left(m\binom{n}{\lfloor \frac{n}{2} \rfloor}\right)$ MIIs.

These proofs also imply that mining MIIs on a rectangular dataset takes exponential time with respect to n in the worst case. Furthermore, these proofs do not just hold for any rectangular dataset, but they also specifically hold for categorical datasets, i.e. datasets in which certain combinations of items cannot co-occur in the same transaction, because they belong to the same category.

In the next chapter, we will conduct experiments using the algorithms described in chapter 2.

Chapter 4

Experiments

In this chapter we evaluate the utility of the different algorithms discussed in chapter 2. To repeat, we developed three algorithms:

1. MII-SUPPRESS(D, θ). This algorithm mines all MIIs on D with support threshold θ, and then generates $\tilde{D}$ by suppressing these MIIs in D. This algorithm offers a guarantee on the frequencies of the rows in $\tilde{D}$ (theorem 2.3).

2. K-MII(D, θ). This algorithm computes $\tilde{D}$ = MII-SUPPRESS(D, θ) and then removes any row $r \in \tilde{D}$ with $sup_{\tilde{D}}(r) \leq \theta$. This algorithm therefore achieves k-anonymity.

3. DP-MII-SUPPRESS(D, θ_1, θ_2, r, β) is a variation on K-MII(D, θ) that offers differential privacy.

MII-SUPPRESS does not offer a conventional privacy guarantee like k-anonymity or differential privacy, although theorem 2.3 does offer a guarantee that comes close to differential privacy. We will therefore primarily use it as a baseline that K-MII and DP-MII-SUPPRESS can be compared against. The goal of our experiments is to research how much utility suffers if we use K-MII instead of MII-SUPPRESS (i.e. if we want a k-anonymity guarantee), and how much utility suffers if we use DP-MII-SUPPRESS (i.e. if we require differential privacy).

We will now discuss which datasets and utility measures were used to measure the performance of these algorithms, and will then present the experimental results

4.1 Datasets, utility measures, implementation details

Non-interactive PPDP algorithms are often evaluated on census data, most commonly on the Adult dataset[6] which is based on the 1994 US Census. However, we run the first two algorithms (MII-SUPPRESS and K-MII) on a wider range of datasets from the LUCS-KDD repository[9]. This repository contains discretized versions of the UCI machine learning repository[23], and it includes a discretized version of the Adult dataset. Besides the Adult dataset, we use the Breast dataset as an example of a medical dataset, which is also a common dataset type for Privacy Preserving Data Publishing. We will also experiment with the PageBlocks dataset. Both the Breast and PageBlocks datasets contain relatively few MIIs per row, which allows us to evaluate the performance of our algorithms on such datasets. Furthermore, we include the chessKRvK dataset in our experiments because it contains many MIIs relative to its size. Lastly, we include the Iris dataset because it is the most popular UCI dataset and it has very low height.

To measure utility, we will measure how many items the algorithms suppress for different parameter choices. In case of K-MII, we will also measure dataset height before and after enforcing k-anonymity for $k = \theta + 1$.

The LUCS-KDD datasets all have a class label, sometimes binary and sometimes multi-class. Therefore we will measure classification accuracy using a random forest classifier. 100 trees are built using bootstrap samples and split quality is measured using Gini impurity. Deciding on the train and test data is not entirely straightforward. Let $D = (R_1, R_2, \ldots, R_m)$ represent the original dataset and let $\tilde{D} = (\tilde{R}_1, \tilde{R}_2, \ldots, \tilde{R}_m)$ represent the suppressed dataset. Both K-MII and MII-SUPPRESS suppress items that are part of an MII. It is possible that for some rows $\tilde{R}_i$, the class label has been suppressed because it was part of an MII. These rows cannot be used for training. Furthermore, K-MII removes infrequent rows after suppression. These rows are also not available for training. Of the remaining rows that *are* available for training, a randomized 80-20 train-test split is performed. The 20% available for testing is combined with a random sample of 20% of the rows that were removed because the class label was missing or because they were infrequent, in order to get a representative sample of the data distribution. Then, as a last step, every row $\tilde{R}_i$ in the test dataset is replaced with its unsuppressed sibling R_i. This way, we train the classifier on sanitized data, but measure its

performance on real, unsanitized data. For every individual experiment, the procedure above (randomized train-test split, training, testing) is executed 100 times and results are averaged.

Experiments were conducted on a computer with an ARM-based M1 chip and 16GB of LPDDR4 RAM. The algorithms were implemented in Rust. As no implementation of any published minimal infrequent itemset mining algorithm could be acquired, we had to implement and test an implementation ourselves. Apriori can very easily be modified to mine all MIIs instead of all FI's, but it turned out to be too slow for larger and denser datasets. Therefore, we implemented the MINIT algorithm[19] in Rust, and actually extended it such that it runs in a multithreaded mode in order to speed up performance.

Some of the datasets have missing values. Our implementation of MINIT computes the most common transaction length and then removes any transaction that does not have this length.

Below, in table 4.1, some descriptive statistics for the datasets that remain can be found, and in table 4.2, the original classification scores can be found.

Dataset	m	n	$\|I\|$	#CL
adult	45222	15	95	2
breast	683	10	16	2
chessKRvK	28056	7	58	18
iris	150	5	19	3
pageBlocks	5473	11	44	5

Table 4.1: Statistics for datasets from the LUCS-KDD repository[9]. For each dataset: height (m), width (n), number of items ($|I|$), and the number of class labels (#CL). These statistics were obtained after removing rows that were incomplete or unusually long.

	adult	breast	chessKRvK	iris	pageBlocks
Accuracy	84.51	94.46	82.69	93.56	92.73

Table 4.2: Mean classification accuracy over 100 runs when trained and tested on original data.

4.1.1 DP-MII-suppress

Our third and last algorithm, the differentially private algorithm DP-MII-SUPPRESS(D, θ_1, θ_2, r, β), performs sampling in order to achieve differential privacy, which makes it unsuitable for small datasets. Therefore, we only use this algorithm on the Adult dataset, which is the largest dataset of the ones mentioned in the previous paragraph. We experiment on DP-MII-SUPPRESS with an ϵ value of 0.1 and δ values of $0.1, 0.01, 0.001$. DP-MII-SUPPRESS offers the differential privacy guarantee in part by using k-anonymity for $k = \theta_2 + 1$. We experiment with k-values of 5 and 10, which means that θ_2 is 4 and 9 respectively. r was fixed to 0.1 and θ_1 and β were chosen following the procedure outlined in section 2.4.2.

4.2 Results

4.2.1 MII-suppress and k-MII

Dataset	$\theta = 1$	$\theta = 5$	$\theta = 10$	$\theta = 50$
adult	7.29	17.30	23.85	46.29
breast	0.48	3.66	6.92	20.46
chessKRvK	94.20	-	-	-
iris	10.26	41.33	48.66	50.00
pageBlocks	0.21	0.39	0.62	1.11

Table 4.3: Percentage of suppressed cells for various θ-values after running MII-SUPPRESS. These results are deterministic. A hyphen indicates that an experiment was not conducted

		$\theta = 1$		$\theta = 5$		$\theta = 10$		$\theta = 50$	
Dataset	m	s	$\tilde{m}$	s	$\tilde{m}$	s	$\tilde{m}$	s	$\tilde{m}$
adult	45222	6.66	44367	16.71	44271	23.23	44038	45.95	42196
breast	683	0.29	680	2.94	668	6.18	666	19.23	655
chess	28056	94.21	28045	-	-	-	-	-	-
iris	150	8.13	145	41.24	145	50.79	139	100.0	150
pBlocks	5473	0.11	5456	0.17	5430	0.28	5408	0.67	5229

Table 4.4: Datasets processed by K-MII. m is the original height of the dataset. K-MII suppresses MIIs and then removes infrequent rows in order to achieve k-anonymity for $k = \theta + 1$. For each θ-value, there are two columns: s represents the percentage of suppressed cells in $\tilde{D}$, and $\tilde{m}$ represents the height of $\tilde{D}$. These results are deterministic. A hyphen indicates that an experiment was not conducted.

		MII-SUPPRESS				K-MII			
Dataset	Orig	$\theta = 1$	$\theta = 5$	$\theta = 10$	$\theta = 50$	$\theta = 1$	$\theta = 5$	$\theta = 10$	$\theta = 50$
adult	84.51	85.16	84.74	84.50	81.58	85.18	84.80	84.51	81.61
breast	94.46	94.67	93.43	90.97	90.68	94.93	93.78	91.18	90.30
chess	82.69	12.10	-	-	-	12.03	-	-	-
iris	93.56	96.77	94.87	95.10	95.03	95.80	95.17	95.13	$\times$
pBlocks	92.73	92.83	92.60	92.53	91.43	92.72	92.43	92.27	90.52

Table 4.5: For different θ values: mean classification accuracy over 100 runs, when trained on datasets generated with MII-SUPPRESS and K-MII, and tested on original data. '$\times$' means that it was impossible to train a classifier due to lack of training data. 'Orig' is the original classification accuracy, taken from table 4.2. A hyphen indicates that an experiment was not conducted

4.2.2 DP-MII-suppress

ϵ	δ	θ_2	β	θ_1	$\lvert\tilde{D}\rvert_{\max}$	$\lvert\tilde{D}\rvert$	s'/s	Accuracy
0.1	0.1	4	0.093	6	3785	3422	13.44 / 43.53	81.69
0.1	0.01	4	0.034	17	1383	1214	14.78 / 60.95	76.10
0.1	0.001	4	0.017	33	691	609	35.60 / 78.01	75.03
0.1	0.033*	9	0.095	12	3866	3405	20.08 / 55.15	80.42
0.1	0.01	9	0.068	17	2767	2525	24.73 / 61.26	77.56
0.1	0.001	9	0.041	27	1668	1477	26.29 / 70.29	76.69

Table 4.6: Mean classification accuracy over 100 runs, when trained on datasets generated with DP-MII-SUPPRESS and tested on original data. DP-MII-SUPPRESS performs partitioning and sampling, yielding a much smaller dataset. The theoretical maximum size of $\tilde{D}$ is equal to original data height ($m = 45222$) times the partition rate ($1 - r$) times the sampling rate (β), and is denoted with $|\tilde{D}|_{\max}$. However, two rounds of k-suppression take place, so the actual size is smaller and denoted with $|\tilde{D}|$. s is the percentage of suppressed cells in $\tilde{D}$ and s' is the percentage of suppressed cells when ignoring rows with a suppressed class label. *Row 4 has $\delta = 0.033$ instead of 0.1, because when $\epsilon = 0.1, \theta_2 = 9$, we get $d(\theta_2 + 1, \beta, \epsilon) \leq 0.033$ for every admissible β.

	Precision	Recall	F1-score	Support
Class 96	0.57	0.49	0.52	2240
Class 97	0.84	0.88	0.86	6805

Table 4.7: Classification metrics for DP-MII-SUPPRESS for $\delta = 0.01$ and $\theta_2 = 4$, i.e. the second entry in table 4.6. Results in this table were obtained from a single run and not averaged over 100 iterations.

	Precision	Recall	F1-score	Support
Class 96	0.00	0.00	0.00	2272
Class 97	0.75	1.00	0.86	6773

Table 4.8: Classification metrics for DP-MII-SUPPRESS for $\delta = 0.001$ and $\theta_2 = 4$, i.e. the third entry in table 4.6. Results in this table were obtained from a single run and not averaged over 100 iterations.

4.3 Discussion

4.3.1 MII-suppress

In table 4.3 we can see how much MII-SUPPRESS suppresses for various datasets and θ values. Firstly, as is expected, the amount of suppression monotonically increases as θ increases. Secondly, MII-SUPPRESS does not significantly modify datasets that have a small number of MIIs. PageBlocks is a very extreme example of this: even when $\theta = 50$, almost 99% of the dataset is left intact. Less extreme datasets with a relatively low number of MIIs, like the medical breast dataset, do not have results quite as spectacular but still yield roughly 80% of intact data for the largest θ value. Similarly, as can be expected, datasets with a very large number of MIIs, like the chessKRvK dataset, are almost completely suppressed even when $\theta = 1$. Experiments with larger θ values were therefore omitted for chessKRvK.

4.3.2 MII-suppress vs k-MII

In table 4.4, we can see how many rows need to be removed after running MII-SUPPRESS in order to achieve k-anonymity for $k = \theta + 1$. Because of theorem 2.3, we expected the number of removed rows to be relatively small. We can see that as θ increases, $\tilde{m}$ decreases for all datasets, but not at an extreme rate. In general, we observe that in our census and medical dataset (Adult and Breast), the number of removed rows is relatively low. This is also true for pageBlocks, which requires barely any suppression because the majority of rows is already part of a frequent equivalence class and therefore does not need any suppression at all. For these three datasets, we can also see that for all θ values, the amount of suppression in a dataset generated by K-MII is actually lower than the dataset generated by MII-SUPPRESS.

This indicates that K-MII tends to remove rows with an above average amount of suppression.

The story is different for chessKRvK (when $\theta = 1$) and for Iris (when $\theta = 50$). In these cases, s is very large (94.21% - 100.0%), yet the ratio $\frac{\tilde{m}}{m}$ remains close to 1. The reason for this is that for these datasets and support thresholds, most rows end up being completely suppressed. As a result, we end up with a large equivalence class of fully suppressed rows. This equivalence class is frequent and therefore k-anonymous. However, the utility of the equivalence class is 0 by any sensible utility metric. Although this happens for chessKRvK and Iris, fully suppressed rows are not common on more realistic datasets like the Adult and Breast.

Classification results for MII-SUPPRESS and K-MII can be found in table 4.5. In general we can see that a classifier trained on a K-MII dataset performs almost identically to a classifier trained on an MII-SUPPRESS dataset. This is consistent with the finding that K-MII generally has to remove only very few rows in order to make the result from MII-SUPPRESS k-anonymous. In fact, accuracy is so close that in a not insignificant number of cases, K-MII based classifiers actually perform ever so slightly better than MII-SUPPRESS classifiers, but this is mostly the result of randomness (although not reported to keep the results section brief, the dispersion between runs is quite often in the range of 2-5%, and at 100 runs this still gives an average that is noisy enough that K-MII can frequently perform slightly better than MII-SUPPRESS).

When looking at accuracy per dataset, we can see that chessKRvK performs very poorly, which is caused by the large amount of suppression. For Adult, Breast and pageBlocks, we see that accuracy slowly drops as θ increases from 1 to 50, but this drop is not more than 4%. For the Iris dataset, accuracy also slowly drops. However, accuracy is consistently higher for the classifiers trained on sanitized Iris data (95.03% - 96.77%), than for the classifier trained on original data (93.56%). We very carefully inspected our code base and performed many reruns because of this anomaly. However, the results are consistent and it is to our knowledge unlikely that they are caused by a bug. We believe the most likely explanation for this anomaly is that the Iris dataset is small (150 rows), so outliers can heavily impact how the classifier is trained. By removing MIIs, the random forest could capture the underlying data distribution better.

Lastly, we observe that there is one case without an accuracy score (Iris for K-MII and

$\theta = 50$). This is because suppression was at 100%, making it impossible to train a classifier. The same likely would have happened to chessKRvK, which is why we did not conduct these experiments to begin with.

4.3.3 DP-MII-suppress

In table 4.6 the results for DP-MII-SUPPRESS can be found. Firstly, we can see that in all cases the ratio $\frac{|\tilde{D}|}{|\tilde{D}|_{\max}}$ is around 0.9, indicating that indeed relatively few rows are removed by the two rounds of k-suppression. This tells us that our data independent generalization scheme (mining MIIs on D_1 with θ_1 and then suppressing these MIIs on D_2) is pretty good at approximating k-anonymity in practice. However, as stated in chapter 2, if the generalization scheme is overly aggressive, then very little k-suppression is required, but the data still has low utility. Therefore, we do not exclusively look at the ratio $\frac{|\tilde{D}|}{|\tilde{D}|_{\max}}$, but also at how much suppression has taken place and how well classification works.

To inspect the degree of suppression, we can compare DP-MII-SUPPRESS with $\theta_2 \in [4, 9]$ to K-MII with $\theta \in [5, 10]$. As we can see in table 4.4, 16.71% of the sanitized Adult dataset was suppressed when $\theta = 5$ and 23.23% when $\theta = 10$. We can see in table 4.6 that the amount of suppression in $\tilde{D}$ is significantly higher when DP-MII-SUPPRESS is used (at least 43.53% when $\theta_2 = 4$ and at least 55.15% when $\theta_2 = 9$. This shows that the generalization scheme is significantly more thorough than is strictly necessary.

Apart from looking at the amount of suppression, we can look at classifier performance. s' tells us how much suppression took place in rows without a suppressed class label. As can be seen, s' is consistently at least twice as small as s, so rows without a suppressed class label er generally less suppressed than average. When looking at classifier accuracy, we can see that accuracy is lower than accuracy on MII-SUPPRESS and K-MII based classifiers (table 4.5), but classifier accuracy is still consistently rather high (75-82%). However, we should note that the Adult dataset is not balanced, and that roughly 25% of the rows contain class label 96 whereas 75% contains label 97. Therefore, there is a risk that at least some of the classifiers have learned to always predict 97. Precision and recall were examined for all 6 classifiers to analyze this. Results for the two worst performing classifiers (row 2 and 3 in table 4.6) can be found in table 4.7 and table 4.8 respectively. As can be seen, the worst performing classifier

indeed only predicts 97, but the second worst does not (neither do the remaining four). The worst performing classifier was also trained on the smallest amount of data, so this is not surprising.

In general, classification accuracy is better for more lenient privacy parameters, because more lenient parameters yield a larger $\tilde{D}$ with less suppression. However, it is interesting to observe the relationship between ϵ and δ on the one hand (the differential privacy parameters) and θ_2 on the other (the k-anonymity parameter). As can be seen, when we fix δ to 0.01 or 0.001 and then increase θ_2 from 4 to 9, many utility metrics actually improve: the sanitized dataset is significantly larger, classifier accuracy is better, and in some cases there is also less suppression. This is because for fixed ϵ and δ, increasing θ_2 allows us to decrease β. This often positively impacts utility because a small θ_2 requires a small β in order to achieve differential privacy, which puts a small upper bound on $|\tilde{D}|_{\max}$. However, the positive impact on utility does not continue indefinitely: if the data curator picks an unrealistically large θ_2, then $|\tilde{D}|_{\max}$ will be large but many rows will be removed during k-suppression.

Chapter 5

Conclusion

We started this work by looking at the general concept of Privacy through Imputation. We have attempted to implement this framework for categorical rectangular datasets by using MIIs. This approach lead to multiple MII-based algorithms with various privacy guarantees, although none of the algorithms ended up being a proper implementation of Privacy through Imputation because imputation was actually not a required source of privacy for any of the implementations.

Our experiments show that mining and imputing MIIs (MII-SUPPRESS) for a certain support threshold θ yields a sanitized dataset that is very close to being k-anonymous (K-MII), where $k = \theta + 1$. For many datasets and θ values, an amount of suppression is applied such that tasks like classification are still possible.

Both these algorithms offer a solid privacy guarantee, but are deterministic and therefore never differentially private. To fully complete the idea of Privacy through Imputation, we attempted to implement imputation in a stochastic manner, but we were unsuccessful. Future work could explore this direction further. A simpler stochastic mechanism was used in combination with our k-anonymous algorithm in order to arrive at a differentially private algorithm (DP-MII-SUPPRESS). This algorithm requires suppression and sampling, which is why it only works on larger datasets. Experiments on the Adult census dataset have shown that it is realistic to generate a differentially private dataset that still has decent utility for a common ϵ value of 0.1. This shows that this algorithm actually works on real world data, although utility is lower compared to the two algorithms that are not differentially private, particularly for

smaller δ values. This confirms the familiar privacy-utility trade-off. Experiments on larger datasets could be conducted in order to see if the algorithm still works for even smaller ϵ and δ values. It could also be interesting to look at more sophisticated methods of developing an MII-based data-independent generalization scheme, rather than just partitioning the dataset into two disjoint parts. Furthermore, it would be interesting to research how the algorithm stacks up against other non-interactive differentially private PPDP methods.

We have not only performed practical experiments. Because of the heavy use of MIIs in our algorithms, we have researched and proved in chapter 3 what the bound on the number of MIIs in a rectangular $m \times n$ dataset is: $m\binom{n}{\lfloor \frac{n}{2} \rfloor} = \Theta\left(m2^{n-\frac{\log(n)}{2}}\right)$. We have shown that for any non-zero n and θ, there exist matrices with this number of MIIs, making the bound tight. This bound is mainly interesting from a theoretical standpoint; real-world datasets will contain significantly fewer MIIs, as has been confirmed by our experiments.

Appendices

Appendix A

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 2 & 2 \\
0 & 1 & 2 & 2 & 0 & 1 \\
0 & 2 & 1 & 2 & 2 & 0 \\
0 & 2 & 2 & 0 & 1 & 2 \\
1 & 0 & 0 & 2 & 2 & 1 \\
1 & 0 & 2 & 1 & 0 & 2 \\
1 & 1 & 1 & 2 & 1 & 2 \\
1 & 1 & 2 & 0 & 2 & 0 \\
1 & 2 & 0 & 1 & 1 & 0 \\
1 & 2 & 1 & 0 & 0 & 1 \\
2 & 0 & 1 & 0 & 2 & 2 \\
2 & 0 & 2 & 2 & 1 & 0 \\
2 & 1 & 0 & 0 & 1 & 1 \\
2 & 1 & 1 & 1 & 0 & 0 \\
2 & 2 & 0 & 2 & 0 & 2 \\
2 & 2 & 2 & 1 & 2 & 1
\end{bmatrix}$$

Table A.1: Smallest matrix with maximum number of MIIs for $n = 6, \theta = 1$. This matrix has dimensions 18×6 and contains $18\binom{6}{3} = 360$ MIIs. This matrix is not an orthogonal array.

$$
\begin{bmatrix}
0&0&0&0&0&0\\
0&0&1&3&1&3\\
0&0&2&1&2&1\\
0&0&3&4&3&4\\
0&0&4&2&4&2\\
0&1&0&2&2&4\\
0&1&1&0&3&2\\
0&1&2&3&4&0\\
0&1&3&1&0&3\\
0&1&4&4&1&1\\
0&2&0&4&4&3\\
0&2&1&2&0&1\\
0&2&2&0&1&4\\
0&2&3&3&2&2\\
0&2&4&1&3&0\\
0&3&0&1&1&2\\
0&3&1&4&2&0\\
0&3&2&2&3&3\\
0&3&3&0&4&1\\
0&3&4&3&0&4\\
0&4&0&3&3&1\\
0&4&1&1&4&4\\
0&4&2&4&0&2\\
0&4&3&2&1&0\\
0&4&4&0&2&3\\
1&0&0&1&3&3\\
1&0&1&4&4&1\\
1&0&2&2&0&4\\
1&0&3&0&1&2\\
1&0&4&3&2&0\\
1&1&0&3&0&2\\
1&1&1&1&1&0
\end{bmatrix}
\begin{bmatrix}
1&1&2&4&2&3\\
1&1&3&2&3&1\\
1&1&4&0&4&4\\
1&2&0&0&2&1\\
1&2&1&3&3&4\\
1&2&2&1&4&2\\
1&2&3&4&0&0\\
1&2&4&2&1&3\\
1&3&0&2&4&0\\
1&3&1&0&0&3\\
1&3&2&3&1&1\\
1&3&3&1&2&4\\
1&3&4&4&3&2\\
1&4&0&4&1&4\\
1&4&1&2&2&2\\
1&4&2&0&3&0\\
1&4&3&3&4&3\\
1&4&4&1&0&1\\
2&0&0&2&1&1\\
2&0&1&0&2&4\\
2&0&2&3&3&2\\
2&0&3&1&4&0\\
2&0&4&4&0&3\\
2&1&0&4&3&0\\
2&1&1&2&4&3\\
2&1&2&0&0&1\\
2&1&3&3&1&4\\
2&1&4&1&2&2\\
2&2&0&1&0&4\\
2&2&1&4&1&2\\
2&2&2&2&2&0
\end{bmatrix}
\begin{bmatrix}
2&2&3&0&3&3\\
2&2&4&3&4&1\\
2&3&0&3&2&3\\
2&3&1&1&3&1\\
2&3&2&4&4&4\\
2&3&3&2&0&2\\
2&3&4&0&1&0\\
2&4&0&0&4&2\\
2&4&1&3&0&0\\
2&4&2&1&1&3\\
2&4&3&4&2&1\\
2&4&4&2&3&4\\
3&0&0&3&4&4\\
3&0&1&1&0&2\\
3&0&2&4&1&0\\
3&0&3&2&2&3\\
3&0&4&0&3&1\\
3&1&0&0&1&3\\
3&1&1&3&2&1\\
3&1&2&1&3&4\\
3&1&3&4&4&2\\
3&1&4&2&0&0\\
3&2&0&2&3&2\\
3&2&1&0&4&0\\
3&2&2&3&0&3\\
3&2&3&1&1&1\\
3&2&4&4&2&4\\
3&3&0&4&0&1\\
3&3&1&2&1&4\\
3&3&2&0&2&2\\
3&3&3&3&3&0
\end{bmatrix}
\begin{bmatrix}
3&3&4&1&4&3\\
3&4&0&1&2&0\\
3&4&1&4&3&3\\
3&4&2&2&4&1\\
3&4&3&0&0&4\\
3&4&4&3&1&2\\
4&0&0&4&2&2\\
4&0&1&2&3&0\\
4&0&2&0&4&3\\
4&0&3&3&0&1\\
4&0&4&1&1&4\\
4&1&0&1&4&1\\
4&1&1&4&0&4\\
4&1&2&2&1&2\\
4&1&3&0&2&0\\
4&1&4&3&3&3\\
4&2&0&3&1&0\\
4&2&1&1&2&3\\
4&2&2&4&3&1\\
4&2&3&2&4&4\\
4&2&4&0&0&2\\
4&3&0&0&3&4\\
4&3&1&3&4&2\\
4&3&2&1&0&0\\
4&3&3&4&1&3\\
4&3&4&2&2&1\\
4&4&0&2&0&3\\
4&4&1&0&1&1\\
4&4&2&3&2&4\\
4&4&3&1&3&2\\
4&4&4&4&4&0
\end{bmatrix}
$$

Table A.2: $\mathrm{OA}_1(5+1,5,3)$, generated using Bush's method. Dimensions are 125×6.

Contains $125\binom{6}{3} = 2500$ MIIs of size 3 when $\theta \leq 4$

Bibliography

[1] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A Inkeri Verkamo. Fast discovery of association rules. *Advances in knowledge discovery and data mining*, 12(1):307–328, 1996.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499, 1994.

[3] Yousef Amar, Hamed Haddadi, and Richard Mortier. An information-theoretic approach to time-series data privacy. In *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*, pages 1–6, 2018.

[4] Roberto J Bayardo and Rakesh Agrawal. Data privacy through optimal k-anonymization. In *21st International conference on data engineering (ICDE'05)*, pages 217–228. IEEE, 2005.

[5] Roberto J Bayardo Jr. Efficiently mining long patterns from databases. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 85–93, 1998.

[6] Barry Becker and Ronny Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[7] Raghav Bhaskar, Srivatsan Laxman, Adam Smith, and Abhradeep Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–512, 2010.

[8] Kenneth A Bush. Orthogonal arrays of index unity. *The Annals of Mathematical Statistics*, pages 426–434, 1952.

[9] Frans Coenen. The lucs-kdd discretised/normalised arm and carm data library (2003). *Software available at* `http://www.csc.liv.ac.uk/~frans/KDD/Software`.

[10] Tore Dalenius. Finding a needle in a haystack or identifying anonymous census records. *Journal of official statistics*, 2(3):329–336, 1986.

[11] Yves-Alexandre De Montjoye, César A Hidalgo, Michel Verleysen, and Vincent D Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3(1):1–5, 2013.

[12] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[13] George T Duncan and Diane Lambert. Disclosure-limited data dissemination. *Journal of the American statistical association*, 81(393):10–18, 1986.

[14] Cynthia Dwork. Differential privacy: A survey of results. In *International conference on theory and applications of models of computation*, pages 1–19. Springer, 2008.

[15] Cynthia Dwork, Nitin Kohli, and Deirdre Mulligan. Differential privacy in practice: Expose your epsilons! *Journal of Privacy and Confidentiality*, 9(2), 2019.

[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[17] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[18] Ashish Gupta, Akshay Mittal, and Arnab Bhattacharya. Minimally infrequent itemset mining using pattern-growth paradigm and residual trees. In *Proceedings of the 17th International Conference on Management of Data*, page 13. Computer Society of India, 2011.

[19] David J Haglin and Anna M Manning. On minimal infrequent itemset mining. In *DMIN*, pages 141–147, 2007.

[20] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2):1–12, 2000.

[21] Håkon Hukkelås, Rudolf Mester, and Frank Lindseth. Deepprivacy: A generative adversarial network for face anonymization. In *International Symposium on Visual Computing*, pages 565–578. Springer, 2019.

[22] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[23] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The uci machine learning repository. `https://archive.ics.uci.edu`.

[24] Diane Lambert. Measures of disclosure risk and harm. *JOURNAL OF OFFICIAL STATISTICS-STOCKHOLM-*, 9:313–313, 1993.

[25] Kristen LeFevre, David J DeWitt, and Raghu Ramakrishnan. Mondrian multidimensional k-anonymity. In *22nd International conference on data engineering (ICDE'06)*, pages 25–25. IEEE, 2006.

[26] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE, 2007.

[27] Ninghui Li, Wahbeh Qardaji, and Dong Su. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pages 32–33, 2012.

[28] Ninghui Li, Wahbeh Qardaji, Dong Su, and Jianneng Cao. Privbasis: Frequent itemset mining with differential privacy. *PVLDB*, pages 1340–1351, 2012.

[29] Jerry Chun-Wei Lin, Yuyu Zhang, Binbin Zhang, Philippe Fournier-Viger, and Youcef Djenouri. Hiding sensitive itemsets with multiple objective optimization. *Soft Computing*, 23:12779–12797, 2019.

[30] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):Article 3, 2007.

[31] Adam Meyerson and Ryan Williams. On the complexity of optimal k-anonymity. In *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 223–228. ACM, 2004.

[32] Aleksander Øhrn and Lucila Ohno-Machado. Using boolean reasoning to anonymize databases. *Artificial Intelligence in Medicine*, 15(3):235–254, 1999.

[33] Stanley RM Oliveira and Osmar R Zaiane. Privacy preserving frequent itemset mining. In *Proceedings of the IEEE ICDM workshop on privacy, security and data mining*, pages 43–54, 2002.

[34] Pierangela Samarati and Latanya Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, SRI International Computer Science Laboratory, 1998.

[35] Xingzhi Sun and Philip S Yu. A border-based approach for hiding sensitive frequent itemsets. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 426–433. IEEE, 2005.

[36] Latanya Sweeney. Simple demographics often identify people uniquely. *Health (San Francisco)*, 671:1–34, 2000.

[37] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[38] Hannu Toivonen et al. Sampling large databases for association rules. In *Vldb*, volume 96, pages 134–145, 1996.

[39] Traian Marius Truta and Bindu Vinay. Privacy protection: p-sensitive k-anonymity property. In *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, pages 94–94. IEEE, 2006.

[40] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.