The Limits of Current Information Loss Metrics as a Measure for Data Utility in k-Anonymization

Thomas Marchand
MSc Student
Departement of Computing
Faculty of Engineering
Imperial College London

ABSTRACT *Collecting and sharing data has been made easier by modern technologies. The number of publicly available datasets increases and with it the risk of privacy infringement. To tackle this issue, several criteria have been proposed to say that a public dataset preserves privacy.. One of them is k-anonymity that imposes that no record can distinguished from at least k-1 other records. To comply with k-anonymity, a dataset is modified, hence decreasing the information it carries. Metrics to evaluate this information loss have been proposed. In this report, our goal is to study those metrics and compare them to the actual drop in performance of a model trained on anonymized data.*

1 Introduction

In the last few decades, collecting data has been made easier. For instance, the number of internet users has been increasing three fold in around ten years [1], the cost of sensors has been divided by two during the same period [2]: we are augmenting the number of ways in which data can be collected. Companies and institutions can create databases related to their activities, clients or environment [3]. For instance TfL, the company that manages London's underground network is now tracking its passengers thanks to the Wi-Fi signals emitted by their mobile phone in the stations. It enables them to better understand and optimize the path finding in stations, to better handle rush hour and maximize the reach of advertising [4] [5]. This outburst of available data is welcome because it comes with scientific discoveries that make use of it. Data is a key element in major recent artificial intelligence breakthroughs [6]. Companies and institutions are now able to take benefits from collected data to optimize and enhance parts of their activities. Data is therefore a strategic component for both companies and institutions as well as the fuel for multiple research sectors [7]. Exchange and publication of data should be beneficial to all actors of society. For TfL, the company is able to sharpen its allocation of resources and boost revenue while passenger enjoy a faster ride from point A to point B in London.

Yet, releasing data as-it-is can be very problematic. A portion of it is personal or related to individuals and releasing it would compromise their privacy. There is a need for disclosure methods that make sure that released data is not breaching individual privacy.

A naive solution to this privacy issue that was and still is commonly used, called pseudonymization, consists in replacing direct identifiers such as names, credit card numbers or social security numbers by pseudonyms. However this proposed solution is not providing enough protection in most cases. As pointed out by L.Sweeney in 2002 [8], if multiple data sets are publicly available it is possible to link some attributes together such as the gender, the date o birth or the marital status between them and uniquely re-identify the records. From a "pseudonymized" medical records of state workers that she received from a state insurance company for research purposes and a voter registration list that she bought, Sweeney showed it was possible to link individuals from one dataset to the other(see figure 1) using the birth date, sex and ZIP code and find out confidential medical information about the then Governor of Massachusetts. She managed to identify the governor because he was the only male with his date of birth living in a 5-digit ZIP code area in both datasets, showing that auxiliary attributes can be used to *re-identify* individuals if they are unique in a dataset. Sweeney introduced *k*-anonymity to tackle this issue.

Since the attack of Sweeney relays on the uniqueness of the target, one possible defence is to make sure that no one is unique. The key idea behind *k*-anonymity is to make sure that it is impossible to distinguish a user from at least *k-1* other distinct users in the database. This ensures that trial of re-identification by linking multiple external databases will always results to groups of at least *k* individuals and not only one like it was the case for the Governor of Massachusetts. Privacy is protected because it is impossible to uniquely identify an individual in the database.However databases are not naturally *k*-anonymous on the contrary, individuals are often unique [9] [10]. Hence databases need to be modified to comply with *k*-anonymity, it can done through suppres-

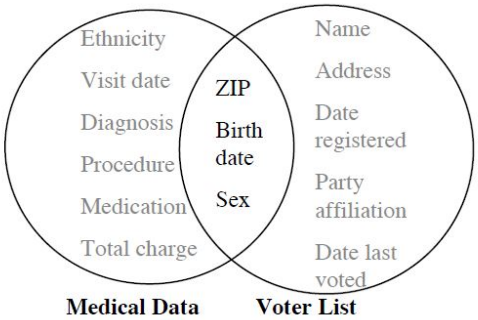


Fig. 1. Figure extracted from Sweeney's [8] showing linkage between two datasets.

Table 1. Fictional financial database

Customer ID	Gender	Age	Balance
1	F	29	250
2	F	24	100
3	M	24	(50)
4	M	24	500
5	N	24	250

Table 2. Global single-dimension generalization

Customer ID	Gender	Age	Balance
1	{F,N}	{24,29}	250
2	{F,N}	{24,29}	100
3	M	{24,29}	(50)
4	M	{24,29}	500
5	{F,N}	{24,29}	250

sions and generalizations. Suppression involves removing records or attributes of the database. Generalization, also called recoding, consists in replacing a value in the table by a set that contains the replaced value, intuitively it is reducing the granularity of the attributes making the data "more similar".

While those database modifications enhance privacy they come with a cost. They damage the data, they lower its precision and diminish the information that is contained in the database. It is a trade-off between privacy and data integrity. Many metrics have been introduced to measure how close to the original dataset a k-anonymized dataset is. Those metrics can evaluate the number of generalizations that have been done [8] [11], if the generalization is too coarse [12] or even the loss of information from a information theory standpoint [13].

However no metric is explicitly related to the final use of the data and how it affects the practical utility of the data. The main contribution of this report is to provide a comparison between known metrics and the actual deterioration in performance of a Machine Learning model caused by anonymization of its training data. This difference of performance is evaluated by comparing models trained on both the original and anonymized datasets. The goal is to assess previously introduced metrics as indicators of the degradation of performance between models. In order to present the metrics used in past papers we summarise previous work from k-anonymity under a harmonised framework.

After presenting the main theoretical concepts, the metrics and the algorithms in sections 2 - 4, we use them in a use case scenario featuring a supervised Machine Learning task to evaluate the metrics in section 5. Sections 6 and 7 include a summary of the study and cover related work.

2 Key Concepts

2.1 Notations:

We are considering tabular databases arbitrarily denoted by the letters T or D. A table has records(rows) and attributes(columns). Without loss of generality we assume that tables have N records and N_A attributes resulting in each table having $N \times N_A$ cells written c_{ij} with $1 \leq i \leq N$ and $1 \leq j \leq N_A$. The set of values of an attribute is noted A_j and is equal to $\{c_{ij}\}_{1 \leq i \leq N}$. We define $\overline{A}_j^{(0)}$ as the set of singletons of A_j's elements. For example if A_j is Gender then $A_j = \{F,M,N\}$ and $\overline{A}_j^{(0)} = \{\{F\},\{M\},\{N\}\}$. Since A_j is a set, we can define its power set $\mathcal{P}(A_j)$ and its cardinality by $|A_j|$. The notation T[$\{A_{j_1},...,A_{j_2}\}$] for $1 \leq j_1 \leq j_2 \leq N_A$ denotes the projection of attributes $\{A_{j_1},...,A_{j_2}\}$ in T. Moreover when a tuple t is considered $t[j]$ is the j^{th} element of this tuple. So if $t = (11,16,19)$ we have $t[2] = 16$.

2.2 Attributes:

The attributes of a tabular dataset that contain information about individuals are of three types [8]. The first type of attributes are *explicit identifiers*, these can alone fully identify an individual within a dataset. Examples would be Names or Social Security Numbers. The second type of attributes are called *quasi-identifiers*[1], they contains general personal information that are not sufficient to alone identify an individual within the dataset but together can be used to reidentify an individual uniquely. In some case, quasi-identifiers can be linked to external publicly available databases to make harmful inferences about individuals(what Sweeney did with Governor). The gender, the date of birth and the marital status are among others often considered quasi-identifiers. Finally, *sensitive attributes* are the ones that relate to sensitive data, these are the attributes that carry useful person-specific information but must not be traced back to the person. Examples would be medical attributes or financial attributes.

Identifying the quasi-identifiers is usually based on domain knowledge and we assume that they are properly defined. For the remaining of this paper except, when explicitly stated, we are considering that pseudonymization has been applied to all datasets such that explicit identifiers have been removed.

[1] Term introduced by Dalenius in [14] and popularized by Sweeney in [8] and [15].

Table 3. Global multi-dimension generalization

Customer ID	Gender	Age	Balance
1	{F,N}	{24,29}	250
2	{F,N}	{24,29}	100
3	M	24	(50)
4	M	24	500
5	{F,N}	{24,29}	250

Table 4. Local multi-dimension generalization

Customer ID	Gender	Age	Balance
1	{F,M}	{24,29}	250
2	{F,M}	{24,29}	100
3	{F,M}	{24,29}	(50)
4	{M, N}	24	500
5	{M, N}	24	250

2.3 *k*-Anonymity:

As demonstrated in the attack done by Sweeney, quasi-identifiers are central to re-identification, we define them formally below.

Definition (**Quasi-identifiers**): the quasi-identifiers of a tabular dataset D is the set of attributes $QI_{\text{D}} \subseteq \{A_1, \ldots, A_{N_A}\}$ whose public release must be controlled [15]. Without loss of generality, we assume that $|QI_{\text{D}}| = N_{QI}$ and that $QI_{\text{D}} = \{A_1, \ldots, A_{N_{QI}}\}$.

Now that quasi-identifiers are defined, we can define *k*-anonymity. A user needs to be indistinguishable from at least *k* others, this means that at least *k* records in a dataset have to share the same quasi-identifiers [8].

Definition (*k*-**anonymity**): D verifies *k*-anonymity if for each record in D[QI_{D}] there is at least *k* similar records in D[QI_{D}], where D[QI_{D}] is the extracted table of D when only considering attributes that are the quasi-identifiers of D.

Taking table 1 as an example: Customer ID is an explicit identifier and its values have been arbitrary reassigned. Gender and Age are quasi-identifiers and we have Balance as a sensitive attribute. Moreover, we can notice that table 1 is not 2-anonymous while table 2 is 2-anonymous. Naturally, *k*-anonymity is defining *equivalence classes*. Records that are indistinguishable are in the same class of equivalence.

Definition (**Equivalence class**): An equivalence class of table T is a set of records that have identical values for the quasi-identifiers QI$_T$ [16]. A table is thus *k*-anonymous if the size of each of its equivalence classes is of at least k.

In table 2 we can define two classes of equivalences: e_1 ={1,2,5} and e_2 ={3,4}, since each class has a cardinalty of at least two, we can state that table 2 is 2-anonymous. We write $\mathcal{E}_{\text{T}}$ the set of all equivalence classes of table T. An equivalence class can be represented by the tuple of quasi-identifers values that is shared by all its records. For an equivalence classe *e* we write e[A_j] the value for the attribute A_j in that tuple. For example in table 2, e_1 can be represented by ({F, M}, {24, 29}) and e_2 by ({M}, {24, 29}) and we also have that $e_1[Gender]$ = {F,M} .

2.4 Generalizations:

In the last two decades a lot of algorithms using different types of generalization have been proposed to achieve *k*-anonymity [17] [18] [19] [20]. To fully understand the difference between each of them, we are proposing a taxonomy of generalizations. Generalizations are simply defining rules to modify the values of a table, they are functions that will take the current set of values of the cells as input and output a bigger set. The core rule of generalization is that the input has to be a subset of the output, this ensures that specific values are replaced by more general, coarsened values. The simplest type is *global single-dimension generalizations*. Those models are operating at the attribute level: attributes are generalized one at a time independently from each others.

Definition (**Global Single-Dimension Generalization**): Let A_j be an attribute of a table T. Let $\overline{A}_j^{(l)} \subseteq \mathcal{P}(A_j)$ and $\overline{A}_j^{(l+1)} \subseteq \mathcal{P}(A_j)$ be two collections of subsets of A_j. $\overline{A}_j^{(l)}$ is the current set of states of the cells for attribute A_j in table T. A mapping $\phi_j : \overline{A}_j^{(l)} \rightarrow \overline{A}_j^{(l+1)}$ is called a Global Single-Dimensional Generalization of attribute A_j if for every $a_{ij} \in \overline{A}_j^{(l)}$, it holds that $a_{ij} \subseteq \phi_j(a_{ij})$. [13]

For example, in a tabular database if the jth attribute is the Age with $A_j = \{11, 16, 19, 24, 24, 31, 34\}$, we could imagine a generalization going from $\overline{A}_j^{(0)}$ to $\overline{A}_j^{(1)} = \{\{11, 16, 19\}, \{24\}, \{31, 34\}\}$ with $\phi_j(\{11\}) = \phi_j(\{16\}) = \phi_j(\{19\}) = \{11, 16, 19\}$, $\phi_j(\{24\}) = \{24\}$ and $\phi_j(\{31\}) = \phi_j(\{34\}) = \{31, 34\}$. We could have two generalizations back to back with the second one being defined from $\overline{A}_j^{(1)}$ to $\overline{A}_j^{(2)} = \{\{11, 16, 19\}, \{24, 31, 34\}\}$. The records with a Age value of 31 would first be generalized to $\{31, 34\}$ then a second time to $\{24, 31, 34\}$. Another example of Global Single-Dimensional Generalization is display using table 1 & 2 where table 1 is a fictional database with two quasi-identifiers : Gender and Age. The two generalization functions are ϕ_A and ϕ_G, they verify $\phi_A(\{F\}) = \phi_A(\{N\}) = \{F, N\}$, $\phi_A(\{M\}) = \{M\}$ and $\phi_G(\{24\}) = \phi_G(\{29\}) = \{24, 29\}$. While table 1 is not 2-anonymous, the generalized table 2 is 2-anonymous.

Global single-dimensional generalizations are defining rules that are applied to all records in a similar fashion. If two records have the same value for an attribute they are generalized in the same way for that attribute, there is a notion of coherence. Moreover it is called Single Dimension because it is operating on only one attribute, indeed we need one generalization function per quasi-identifier. To gain more flexibility, the generalization can be done on multiple attributes at the same time. It is then called *Global Multi-Dimension Generalization*. In this case, only one generalization function is needed for the whole table. The principle is the same as above but now the input is a tuple containing the values for each of the quasi-identifier.

Definition (**Global Multi-Dimension Generalization**): Let $A_1,...,A_{N_{QI}}$ be quasi-identifiers of a table T. Let $\overline{A}_j^{(l)} \subseteq \mathcal{P}(A_j)$ and $\overline{A}_j^{(l+1)} \subseteq \mathcal{P}(A_j)$ for $1 \leq j \leq N_{QI}$ be collections of subsets of each quasi-identifiers. $\overline{A}_j^{(l)}$ is the current set of states of the cells for attribute A_j in table T.
A mapping $\phi : \overline{A}_1^{(l)} \times ... \times \overline{A}_{N_{QI}}^{(l)} \rightarrow \overline{A}_1^{(l+1)} \times ... \times \overline{A}_{N_{QI}}^{(l+1)}$ is called a Global Multi-Dimensional Generalization of the table T with respect to its quasi-identifiers if for every $(a_{i1},...,a_{iN_{QI}}) \in \overline{A}_1^{(l)} \times ... \times \overline{A}_{N_{QI}}^{(l)}$, it holds that $a_{ij} \subseteq \phi(a_{i1},...,a_{iN_{QI}})[j]$ for $1 \leq j \leq N_{QI}$. [13]

An example of global multi-dimensional generalization is displayed table 3. The function ϕ is taking tuple this time. $\phi((\{F\},\{29\})) = \phi((\{F\},\{24\})) = \phi((\{N\},\{24\})) = (\{F,N\},\{24,29\})$ and $\phi((\{M\},\{24\})) = (\{M\},\{24\})$. There is a unique function for the whole table, it takes tuples as input and outputs tuples and for identical input tuples the output will be the same. Multi-dimension generalization are more flexible compared the single-dimension because every single-dimension generalization can be expressed as a multi-dimension generalization [16] but the reverse statement is false. Indeed starting from table 1 it is impossible to obtain table 3 by only using single-dimension generalization because the value 24 is sometime generalized into $\{24,29\}$ and sometime left untouched depending on its value for the attribute Gender. All the above cases are global generalizations.

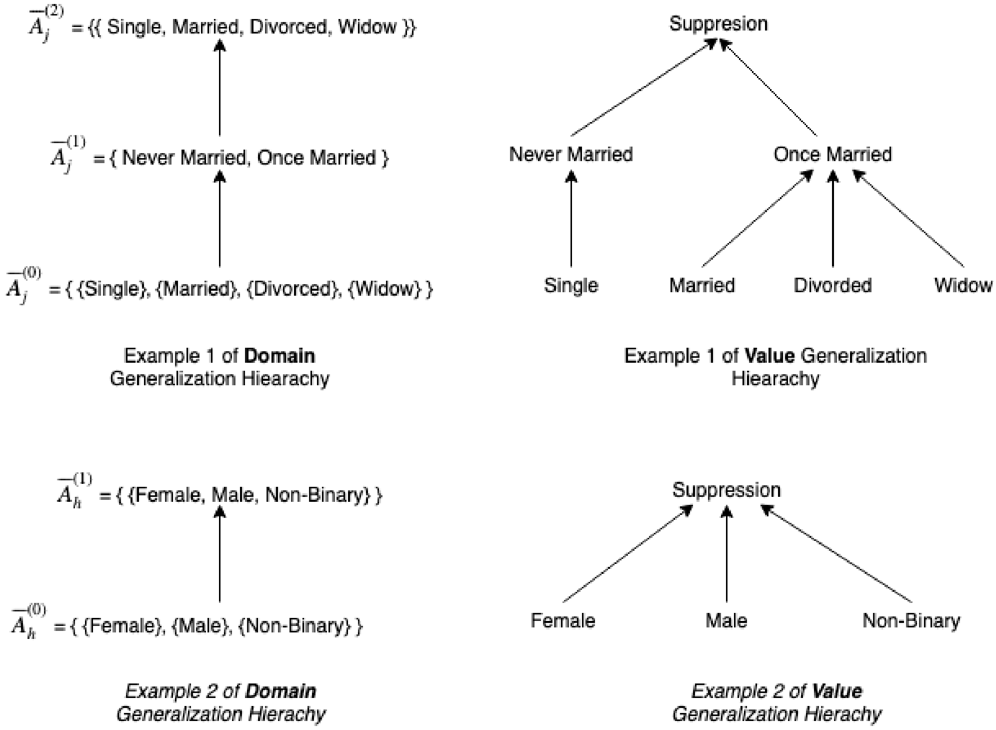
Local generalizations are functions that can, for the same input have a different output. It is more flexible than global generalizations but it is less intuitive and more difficult to compute and interpret, hence less popular. An example of local generalization is shown table 4. Records 3 and 4 that have identical quasi-identifiers are generalized differently. Similarly global models can always be expressed as local models but the reversed statement is not always true [16]. Multiple local generalization models have been presented in [21] [22] [20], their impact on information loss metrics is limited so we are focusing on global generalizations.

From a practical stand point, we still need to determine how to construct the generalization functions. Generalizations can be either *Hierarchy-based* or *Partition-based*. Hierarchy-based generalization are used for categorical attributes and requires *Domain and Value Generalization Hierarchies* for the attributes. The *Domains* are defined by [15] as the possible set of values of an attribute at a given time. The *Domain Hierarchies* define the sequence of domains that can be obtained when multiple successive generalizations are applied to an attributes. Formally the Domain Generalization Hierarchies of attribute A_j written DGH$_j$ is actually the list of the codomains $[\overline{A}_j^{(0)},\overline{A}_j^{(1)},...,\overline{A}_j^{(p)}]$ of the generalization functions where p is the number of generalizations. Each domain corresponds to a generalization step. We also define $|\text{DGH}_j| = p-1$ as the height of the domain generalization hierarchy. Following our example that was started at the beginning of this section, we have A_j being the Age, the Domain generalization hierarchy is $\text{DGH}_j = [\overline{A}_j^{(0)},\overline{A}_j^{(1)},\overline{A}_j^{(2)}]$ with $\overline{A}_j^{(0)}$ being the singleton set previously defined, $\overline{A}_j^{(1)} = \{\{11,16,19\},\{24\},\{31,34\}\}$ and $\overline{A}_j^{(2)} = \{\{11,16,19\},\{24,31,34\}\}$. We have $|\text{DGH}_j| = 2$

Value Generalization Hierarchies characterize how the attributes' values are related from one domain level to the upper next. It details how to practicly handle the generalization. It is usually manually defined because it may require prior knowledge related to the specific environment of the dataset. Generalization Hierarchies are mostly applied in the global single-dimension cases because they are simple and can be represented by trees. An example taken from [15] where A_j corresponds to the Marital Status can be seen on Figure 2. The initial domain $\overline{A}_j^{(0)}$ is {{Single}, {Married}, {Divorced}, {Widow}}. It can be generalized once into the new domain $\overline{A}_j^{(1)}$ = {Never Married, Once Married}. With Never Married = {Single}, and Once Married = {Married, Divorced, Widow} in order to lighten the writing but if we want to explicitly write it we end up with $\overline{A}_j^{(1)}$ = {{Single},{Married, Divorced, Widow}}. The left part of the figure displays the succession of domains for the attribute A_j, it is the Domain Generalization Hierarchy. The right part of the figure explicit how to do the generalization in practise with the mapping of the values from one domain to the next one, it is the Value Generalization Hierarchy.

Partition-based generalization are applied to attributes that lay in a totally-ordered domain, most of the time numerical attributes. This type of generalization consists in partitioning the domain space into intervals and each value contained in the interval is replaced by the interval itself. This method is used because of its simplicity. Instead of manually defining hierarchies, the domains are just partitioned into intervals. An example would be to provide an age range of a person rather than their exact age. So following our example with A_j being the age, we have $A_j = \{11,16,19,24,24,31,34\}$, the new domain could be $\overline{A}_j^{(1)} = \{[10-19],[20-29],[30-39]\}$.

The definition of *suppression* varies from one paper to another. It can be defined as the ultimate generalization: the attribute is generalized into a domain that does not discriminate any of the records, some papers refer to this extreme case of generalization as a *generalization by suppression* [23] [13]. All the records have the same value that is the set of all possible values, in our example from figure 2 : {{Single, Married, Divorced, Widow}}. The first domain of a domain generalization hierarchy is always the set of singletons and the last domain is always one unique set containing all possible values. Other type of suppressions are considered in the literature. Record suppression consists in simply removing records of the database [11] [15]. Local suppressions as described by T.De Waal and L.Willenborg in [20] consists in replacing some cell values by missing values.


$\overline{A}_j^{(2)}$ = {{ Single, Married, Divorced, Widow }}
$\overline{A}_j^{(1)}$ = { Never Married, Once Married }
$\overline{A}_j^{(0)}$ = { {Single}, {Married}, {Divorced}, {Widow} }
Suppresion
Never Married
Once Married
Single
Married
Divorded
Widow
$\overline{A}_h^{(1)}$ = { {Female, Male, Non-Binary} }
$\overline{A}_h^{(0)}$ = { {Female}, {Male}, {Non-Binary} }
Suppression
Female
Male
Non-Binary

Example 1 of **Domain** Generalization Hieärachy

Example 1 of **Value** Generalization Hieårachy

*Example 2 of **Domain** Generalization Hierachy*

*Example 2 of **Value** Generalization Hierachy*

Fig. 2. Examples of hierarchies in hierarchy-based generalizations inspired from Sweeney's [15].

3 Metrics

The taxonomy from the previous section enables a classification of most of the methods that modify a dataset to achieve *k*-anonymity through generalizations. Metrics were introduced to evaluate the dataset degradation resulting from those modifications.

3.1 Hierarchy-based Metrics:

Sweeney and Samarati [15] introduce the notion of *distance* to characterize the level of generalization of a table with respect to another table in the case of single-dimension hierarchy-based models. The *absolute distance* as defined in [15] is simply the number of generalization steps that are needed to go from one table to the other. It is summed over all the quasi-identifiers. For example if we are trying to compute the absolute distance between the original table T and one table T' obtained using hierarchy-based generalizations we apply the following formula.

$$\mathrm{Absdist}(\mathrm{T},\mathrm{T}') = \sum_{j=1}^{N_{QI}} d_j \tag{1}$$

where d_j is the number of times that attribute A_j has been generalized i.e. the height of his domain in his Domain Generalization Hierarchy. This metric has one big issue: every generalization is equally penalized regardless of the attribute. Depending on the hierarchies some generalizations may heavily deteriorate the data while some have minimal effects, Absdist penalizes them in a same way. The *relative distance* corrects this by weighting each term of the summation by the total height of its generalization hierarchy. The formula to compute the relative distance between an original table T and one of its generalized table T' is below.

$$\mathrm{Reldist}(\mathrm{T},\mathrm{T}') = \sum_{j=1}^{N_{QI}} \frac{d_j}{|DGH_j|} \tag{2}$$

It can be seen as a normalized version of the absolute distance where each attribute is penalized according to its relative generalization by being given a score between 0 (not generalized at all) and 1 (generalized by suppression). Those metrics are limited to single-dimension generalizations because we are summing over attributes, to adapt the last metric to multi-dimension models we have to sum over all cells. Sweeney also presented the *precision* score in [11] with the idea of summing over all cells and normalizing the whole sum.

$$\mathrm{Prec}(\mathrm{T},\mathrm{T}') = 1 - \frac{\sum_{j=1}^{N_{QI}} \sum_{i=1}^{N} \frac{h_{ij}}{|\mathrm{DGH}_j|}}{N_{QI} \times N} \tag{3}$$

with h_{ij} being the number of times that the value of the i^{th} record and j^{th} attribute has been generalized. Prec looks at how each cell has been generalized compared to the height of the domain generalization hierarchy. Generalizations on short domain are more penalized compared to generalizations on deeper domains. It is important because precision tells us in average what percentage of generalization hierarchies have been used which is quite insightful.

Despite being intuitive, those metrics share multiple downsides. First, they do not give real insight about the utility of the data. Counting the number of generalization steps is a bit naive to evaluate the degradation of the data. Second, those metrics are heavily dependent on the hierarchies used hence it is impossible to compare generalizations that use different hierarchies. Since we don't have methods to evaluate the hierarchies by themselves those metrics are clearly limited. Finally, as a consequence of the second point, they can only be applied to hierarchy-based models. This pushed researchers to seek for other means of measurements.

3.2 Discernibility Metric:

R.Agrawal and R.J.Bayardo proposed the *discernibility metric* (DM) in [12] to give an idea of how optimized are the size of the equivalence classes induced. Coarse generalization may lead to bigger classes than wanted. When seeking k-anonymity, ideally we want equivalent classes of size k and not bigger because it would mean that we generalized more than needed. Each record is penalized with the value of the size of its equivalence class. Once the table T is k-anonymous we can compute the discernibility using the following definition [16].

$$\mathrm{DM(T)} = \sum_{\forall e \in \mathcal{E}_\mathrm{T}} |e|^2 \tag{4}$$

This simple metric is giving us information about the quality of the anonymization that has been done. Yet it says nothing about how the data has been modified or if it is still useful or not. Moreover it is not very discriminating, lots a different table obtained with different successive generalization have the same discernibility. For example table 2 and 3 are 2-anonymous and have the same discernibility value even though 2 has been over generalized compared to table 3. All of the above make the discernibility metric a good and interesting side metric to determine if the anonymization has been too gross but it is not telling enough to be the main metric.

3.3 Normalized Certainty Penalty:

Another metric that can be used in all cases is the *normalized certainty penalty (NCP)* introduced by J.Xu et al in [24]. This metric quantifies the information loss of an equivalence class associated to an attribute by measuring the fraction of the initial domain values that have been generalized [25]. That is the bigger the cardinality of a value the bigger the penalty. For a hierarchy-based model this corresponds to dividing the cardinality of the current value shared by the whole equivalence class by the cardinality of the initial domain.If the value is untouched we set the value of the *NCP* to zero. The *NCP* for an equivalence class e for a attribute A_j in the case of a hierarchy-based model is given below [26].

$$NCP_{A_j}(e) = \begin{cases} 0, & |e[A_j]| = 1 \\ \frac{|e[A_j]|}{|\overline{A}_j^{(0)}|} & \text{otherwise} \end{cases} \tag{5}$$

For a partition-based model, it corresponds to dividing the current range of values by the initial range of values. The *NCP* for an equivalence class e for a numerical attribute A_j in the case of a partition-based model is given below [26].

$$NCP_{A_j}(e) = \frac{\max(e[A_j]) - \min(e[A_j])}{\max(\overline{A}_j^{(0)}) - \min(\overline{A}_j^{(0)})} \tag{6}$$

Since we are only computing the *NCP* for an equivalence class relatively to only one attribute, to compute the *NCP* of the whole table we have to sum over equivalence classes and over quasi-identifiers. Attributes are weighted by a factor w_j in case there is preferences between them, normalization is also applied similarly to [26] to bring it back between 0 and 1.

$$NCP(T) = \frac{1}{N \times N_{QI}} \sum_{\forall e \in \mathcal{E}_\mathrm{T}} \sum_{j=1}^{N_{QI}} w_j \cdot NCP_{A_j}(e) \tag{7}$$

This metric is focusing on the fact we lose information by replacing "Married" by "Once Married"(see figure 2) which is a set of 3 possible values. A value of 0 means no generalization an a value of 1 means that all records are fully generalized. The main advantage of this metric is that it is intuitively penalizing the loss of granularity in the data. Moreover this metric can be used in any type of generalization and does not depend on the hierarchies. This metric can be applied to tables generalized from different hierarchies. More complete compared to previous metrics, it still does not take into account the utility of the data. Is loss of certain part of the data more important to the task at hand ?

3.4 Classification Metric:

Classification Metric (CM) proposed by V.Iyengar in [27] is an attempt to answer this question in the case where the data is aimed at training a classification model. The data has an attribute label that is considered a sensitive attribute so it is not generalized and it is left untouched. This label is the attribute that we want to predict in the classification task. The key idea of *CM* is to predict the classification error that is resulting from the generalization. Once the data is anonymized, each equivalence class is given the label that is shared by the majority of its records. Each record now has a label and the label of its equivalence class. If the two are not identical a penalty of one is applied. Formally it can be described as follows.

$$CM(\mathrm{T}) = \sum_{i=1}^{N} penalty(r_i) \quad (8)$$

with $penalty(r_i) = 0$ if the label of r_i is the majority label in the equivalence class of r_i and $penalty(r_i) = 1$ if they are different. The idea is that if they are not identical it is probable that the classifier is going to misclassify it. This metric would be very accurate if all attributes were quasi-identifiers. Intuitively, the CM is the error rate of a k-NN classifier tested on the training set, it is labeling each data point with the label of its class. Similarly to DM this metric can be used as an auxiliary metric to assess if the anonymization has been done in a coherent way considering that the data is going to be used in a classification.

3.5 Entropy:

Information theory provides useful tools to quantify information loss. Entropy is a known method to compute uncertainty. Introduced by L.Willenborg and T. de Waal in [20] and [28] in the context of k-anonymity then re-used more recently by A.Gionis et al. in [13], the conditional entropy can directly compute the amount of information needed to describe the original table from the generalized one i.e. the information lost during the generalization process. Indeed, $\mathrm{H}(Y|X)$ quantifies the amount of information needed to describe the outcome of a random variable Y given that the value of another random variable X is known [29]. For example, if after some generalizations, we know that the gender of an arbitrary record is in $\{\mathrm{M}, \mathrm{F}\}$, the additional information needed to figure out if the record's gender is actually a M or a F is given by $\mathrm{H}(V|V \in \{\mathrm{M},\mathrm{F}\})$ with V begin the random variable describing the value of the record's gender. Given a table T and its generalization T', for the i^{th} record and the j^{th} attribute we write a_{ij} the value of the cell in T'(which is a set) and c_{ij} the value of the cell T, the information loss for this cell is therefore given by the computation of the conditional entropy [13].

$$H(V_{ij}|a_{ij}) = -\sum_{v \in a_{ij}} \mathrm{P}(V_{ij} = v|a_{ij}) \log \mathrm{P}(V_{ij} = v|a_{ij}) \quad (9)$$

with V_{ij} being the value of the attribute A_j for the i^{th} record. The conditional probabilities are established by counting occurrences in the initial table T [13].

$$\mathrm{P}(V_{ij} = v|a_{ij}) = \frac{|\{1 \leq i \leq N : c_{ij} = v\}|}{|\{1 \leq i \leq N : c_{ij} \in a_{ij}\}|}, v \in a_{ij} \quad (10)$$

Since equation 9 is only for one cell, we need to sum over all the records and all the attributes to obtain the entropy of generalizing T into T' [13].

$$\Pi_{\mathrm{e}}(\mathrm{T},\mathrm{T}') = \sum_{i=1}^{N} \sum_{j=1}^{N_{QI}} H(V_{ij}|a_{ij}) \quad (11)$$

The main strength of this metric is its information theory background, it has been widely use to quantify information loss [30]. Yet, entropy is still difficult to interpret in the context of anonymization especially how it can give insights about the data utility.

4 Algorithms:

In this sections we are exploring different algorithms that have been proposed to achieve k-anonymity. We identify which generalization model they use and if they provide an optimal solution with respect to a metric or an approximation of it.

4.1 Datafly and MinGen:

Datafly and MinGen have been introduced by Sweeney in [11] [31]. MinGen offers a naive but optimal(with respect to the precision metric) method to find k-anonymous tables while Datafly offers a approximate and practical way to reach k-anonymity.

First, for both algorithms, domain and value generalization hierarchies(DGHs & VGHs) need to be constructed for every quasi-identifiers of the database. Similarly to what has been done in figure 2, each attributes has a tree describing its potential generalizations.

Sweeney's definition of generalization is compatible with what is presented in this paper. In addition to it, she defines a partial order, $\geq$ and $\leq$ between tables. This binary relation can be translated to words as "is more/less generalized than", formally T $\leq$ T' means that we can obtain T' by generalizing T. For example : table 1 $\leq$ table 3 $\leq$ table 2 and table 1 $\leq$ table 4.

Since an attribute A_j can be generalized $|DGH_j|$ times, it can occupy $|DGH_j| + 1$ states therefore if we consider all the quasi-identifiers of a database we have

$$\prod_{j=0}^{n} (|DGH_j| + 1) \quad (12)$$

possible combinations of generalizations [11] that can lead to a k-anonymous table with $|DGH_j|$ the height of the domain generalization hierarchies defined 2.4. Among all those tables only a few are going to be of interest for us, the k-anonymous ones that are generalization of non k-anonymous tables [11].

Definition (k-**minimal generalization**): Let's $\mathrm{T_i}$ and $\mathrm{T_j}$ be two tables such that $\mathrm{T_j}$ is a generalization of $\mathrm{T_i}$ i.e. $\mathrm{T_i} \leq \mathrm{T_j}$. The table $\mathrm{T_j}$ is said to be a k-minimal generalization of the table $\mathrm{T_i}$ with respect to their common quasi-identifiers if and

only if:
1. T_j is k-anonymous.
2. $\forall T_z : T_i \leq T_z, T_z \leq T_j, T_z$ is k-anonymous $\Longrightarrow T_z = T_j$.

This corresponds to generalizing the initial table into k-anonymous tables that are not generalized more than needed. This definition is a first effort in trying to keep the data integrity as high as possible. Yet again, among those k-minimal generalizations not all are equally useful. Sweeney uses *Precision*(defined in 3.1) to select the best tables among the k-minimal generalizations [11].

Definition (k-**minimal distortion**): Let's T_i and T_j be two tables such that T_j is a generalization of T_i i.e. $T_i \leq T_j$. T_j is said to be a k-minimal distortion generalization of the table T_i with respect to their common quasi-identifiers if and only if:
1. T_j is k-anonymous.
2. $\forall T_z : Prec(T_i) \geq Prec(T_z), Prec(T_z) \geq Prec(T_j), T_z$ is k-anonymous $\Longrightarrow T_z = T_j$.

From the definition above it is possible to design a naive algorithm that outputs a k-minimal distortion table. Sweeney proposed the MinGen algorithm in [11]. The main idea is to compute all possible generalized tables then to find the ones that are k-anonymous among them. Once the k-anonymous solution have been isolated, a search for the minimum for the *Prec* metric gives out one or potentially multiple k-minimal distortion tables.

Algorithm 1 Preferred Minimal Generalization : MinGen

```
Require: T a table with already defined quasi-identifiers
    and DGH for each of them. $k$ fixed integer. A function
    $preferred()$ to choose among the multiple solutions.
Ensure: $|T| > k$
  if T already $k$-anonymous then
      MGT ← T
  else
      $allgen \leftarrow \{ T_i : T_i$ generalized table of T$\}$
      $protected \leftarrow \{T_i : T_i \in allgen$ and $T_i$ $k$-anonymous $\}$
      $MGT \leftarrow \arg\max_{T_i \in protected} Prec(T_i)$
      $MGT \leftarrow preferred(MGT)$
  end if
  return MGT
```

Given the above definitions, MinGen outputs optimal solutions in a sense that it is optimized for the *Prec* metric. It penalizes heavily short DGHs compared to long DGHs(because of the $\frac{1}{|DGH_i|}$). One has to keep this in mind when designing the DGHs and VGHs. This algorithm suffers from the drawbacks of the *Precision* metric. In addition, MinGen is difficult to use in real-life cases where the number of quasi-identifiers is high and where DGHs and VGHs are complicated. The equation 12 shows that it can quickly become unpractical to do extensive searches in the space of the potential generalized table. According to Sweeney herself: "With respect to complexity, MinGen makes no claim to be efficient" [11], she proposed the Datafly algorithm to tackle the complexity issue. Indeed according to [23], an optimum anonymization probel that guarantees k-anonymity while minimizing an information loss function is NP-Hard. Datafly uses heuristics to make k-anonymity practical.

Datafly [31] [11] uses a global single-dimension hierarchy-based generalization model. The main idea behind it is to iteratively generalise individual attributes to reach k-anonymity, by selecting greedily the attribute with the highest number of distinct values. Datafly does not compute all generalization tables to select the best one (what MinGen does) but instead starts from the initial table and iteratively applies single-dimension generalizations to the attributes that have the biggest domain until the table is k-anonymous given a few suppressions. When all but $p \leq k$ records are in an equivalence class of size k or more, the p records are suppressed i.e. not released. Datafly's main tool is the $freq$ table that is built at the beginning of each iteration: it is a table containing the distinct tuples from $T[QI_T]$, along with their number of occurrences [11]. Frequency tables are easily(but costly) built with a COUNT combined with a GROUP BY sequence in SQL. The process of the Datafly algorithm is described in figure 3 and the pseudo code is also available, see algorithm below.

Algorithm 2 Datafly algorithm

```
Require: T table with already defined quasi-identifiers and
    DGH for each of them. $k$ fixed integer. A function
    $preferred()$ to choose among the multiple solutions.
Ensure: $|T| > k$
  $freq \leftarrow$ frequency table that contains the number of oc-
  currences of each distinct tuple of $T[QI]$
  while more than $k$ records have distinct tuples in $freq$
  i.e. suppressing $k$ well-choosen records is not enough to
  achieve -anonymity. do
      $A \leftarrow$ the attribute in freq having the highest number of
      distinct values
      $freq \leftarrow freq$ with attribute $A$ generalized once more.
  end while
  $freq \leftarrow freq$ with the distinct tuples suppressed.
  DT ← rebuild record table from $freq$ table.
  return $preferred$(DT)
```

The choice of the generalized attribute is done by a heuristic. Indeed, choosing the attribute based on the size of the domains is not always the best option. Some common quasi-identifiers such as Marital Status and Gender typically have small domains while Age or ZIP code have much bigger domains, this induces an a priori order on the generalizations that is not necessarily wanted. This may lead to a high number of inefficient generalizations and non optimal solution: Datafly guarantees k-anonymity but only gives an approximation of the optimal solution.

Nevertheless Datafly has been heavily used in the industry to anonymize clinical datasets [32] [33] thanks to its

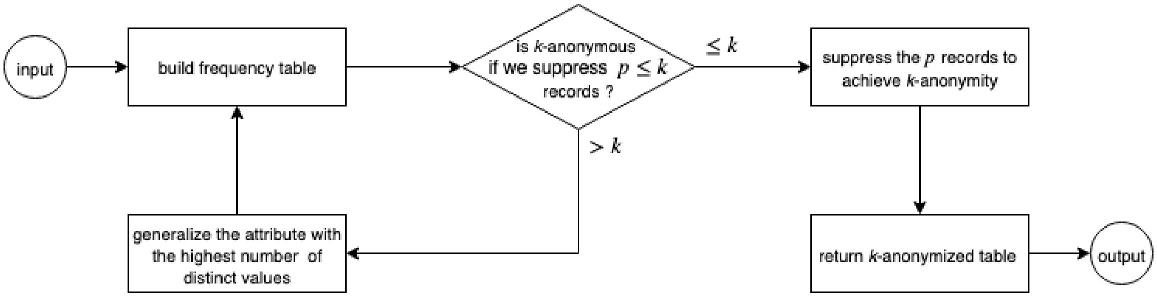

input
build frequency table
is k-anonymous if we suppress $p \leq k$ records ?
$\leq k$
suppress the p records to achieve k-anonymity
$> k$
generalize the attribute with the highest number of distinct values
return k-anonymized table
output

Fig. 3. The Datafly algorithm to achieve k-anonymity.

computational efficiency and its guarantee of result.

In the following years, many algorithms relying on domain and value generalization hierarchies were presented [34] [19] [24] [35] [27].

4.2 Incognito:

Presented by K.LeFevre, D.J.DeWitt and R.Ramakrishnan in 2005, [19], Incognito is a global, single-dimension and hierarchy-based model. Therefore each attribute has its DGH and VGH already defined. A nice way to represent the generalization possibilities that are offered by those hierarchies for a given database is to draw its *generalization lattice*, also called *strategy* [15] [19] [36].

Given a tabular database with two quasi-identifiers: B for Birthdate and G for Gender, let's imagine that their domain hierarchies are $\text{DGH}_B = [B^{(0)}, B^{(1)}, B^{(2)}]$ and $\text{DGH}_G = [G^{(0)}, G^{(1)}]$. The generalization lattice is represented in figure 4 (a) and the corresponding vector lattice in (b). Each arrow represents one generalization step for one attribute only. Each node is a possible table that is a generalization of the initial table into the combination of domains displayed in the node. For example, the node $(B^{(1)}, G^{(0)})$ is representing the table after attribute B has been generalized once according to its hierarchies. The number of node is computed with equation 12. The bottom node is the initial table and the top node is the table where every attributes has been totally generalized.

A bottom-up search of this graph checking for the k-anonymity of each node could reveal which generalizations of our table are k-anonymous.

Incognito uses three domain generalization hierarchy properties to create an algorithm that is viable and optimal according the order relation defined by Sweeney presented in section 4.1. They are straightforward and key to Incognito and many future algorithms.

Property (**Generalization**): Let T be a table. If T$'$ and T$''$ are two generalizations of T such that T$' \leq$ T$''$ then we have : T$'$ k-anonymous $\Longrightarrow$ T$''$ k-anonymous [19].

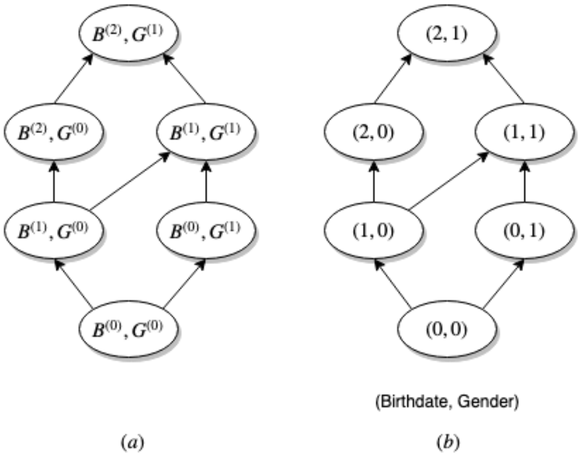


Fig. 4. Example of lattices inspired from Sweeney's [15]

This simply means that if $\text{D}_{<B^{(1)},G^{(1)}>}$ is k-anonymous then $\text{D}_{<B^{(2)},G^{(1)}>}$ is also k-anonymous similarly if $\text{D}_{<B^{(1)},G^{(0)}>}$ is not k-anonymous then we can infer that $\text{D}_{<B^{(0)},G^{(0)}>}$ is not as well. $\text{D}_{<B^{(2)},G^{(1)}>}$ being the table D where the quasi-identifier B has been generalized twice and now lies into the domain $B^{(2)}$ and where G has been generalized once into $G^{(1)}$.

Property (**Rollup**): Let T be a table. If T$'$ and T$''$ are two generalizations of T such that T$' \leq$ T$''$ then computing the frequency table F$''$ of T$''$ given the frequency table F$'$ of T$'$ is simple: we sum each set of counts in F$'$ that gets generalized to the same value in T$''$ and write it down in F$''$ [19].

Very intuitively, when looking at figure 2, the above property implies that the number of records that have 'Once Married' as Marital Status is equal to the sum of the number of records that previously had 'Married' or Divorced' or 'Widow'. This property sheds light on a little trick that reduces the cost of updating a frequency table after a generalization. This property is analogous to the flow conservation of graph theory.

Property (**Subset**): Let T be a table. If T is k-anonymous with respect to QI_{T} then it is also k-anonymous with respect to any set of attributes S that is a subset of QI_{T}, i.e. $S \subset QI_{\text{T}}$ [19].

This property implies that if $\text{D}_{<B^{(1)},G^{(0)}>}$ is k-anonymous we know that $\text{D}_{<B^{(1)}>}$ and $\text{D}_{<G^{(0)}>}$ are also k-anonymous. We can also revert it and deduce that if $\text{D}_{<B^{(0)}>}$ is not k-anonymous then $\text{D}_{<B^{(0)},G^{(0)}>}$ or $\text{D}_{<B^{(0)},G^{(1)}>}$

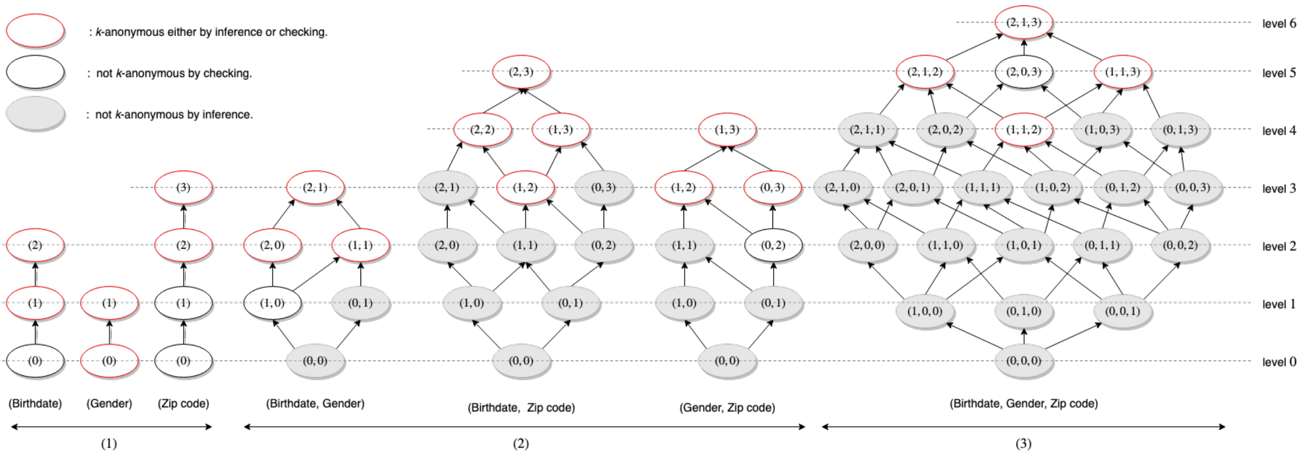


Fig. 5. Incognito algorithm with three quasi-identifiers detailed by iteration inspired from Kohlmayer et al. [36]

are not either.

The Incognito algorithm finds the set of all possible k-anonymous generalizations of a table by tagging as k-anonymous or not the nodes of its generalization lattice. The main idea behind it is to use the Generalization and Subset properties to do predictive tagging. Using dynamic programming [19] [36], it starts by considering the k-anonymity of the table with respect to only single-attribute subsets of the quasi-identifiers and then, with respect to subsets containing two attributes, and so on until all the quasi-identifiers are considered at once. Between each iteration the intermediate generalization lattices are pruned with the Generalization and Subset properties that enable predictive tagging from the lattices of the previous iteration. The k-anonymity of the rest of the nodes(where predictive tagging does not apply) are established with a bottom-up breadth-first search of the lattice by building the frequency tables. At the end, when all quasi-identifiers are considered, the generalization lattice is heavily pruned and the search space is greatly reduced.

By iteratively increasing the size of the subsets of attributes considered and searching their lattice, Incognito is capitalizing on previous taggings to do predictive tagging(thanks to the properties) on the following ones to finally end up considering all the quasi-identifiers with a simplified lattice. Once the final simplified lattice is searched we have the set of all possible k-anonymous generalizations.

With the Generalization property we can infer the k-anonymity status of some nodes given the status of related nodes in the same lattice. Indeed all the parent nodes of a non k-anonymous node are non k-anonymous and every child of k-anonymous node is k-anonymous. It speeds-up the processing time of a lattice. The Subset property enables an a-priori pruning of the lattices at iteration i given iteration $i-1$. Nodes that are extension of nodes previously ruled out as non k-anonymous nodes can be directly flagged as non k-anonymous. This is used to reduce the size of the lattice hence reducing the search space for each iteration. For example at iteration 1 we check $D_{<B^{(0)}>}$ and find that it is not k-anonymous, at iteration 2 we can infer that $D_{<B^{(0)},G^{(0)}>}$ and $D_{<B^{(0)},G^{(1)}>}$ are not k-anonymous without doing any computations. This can be seen in figure 5 when nodes are colored in grey.

An example of Incognito applied to a table with three quasi-identifiers is given in figure 5. It is detailed iteration by iteration; since we have three quasi-identifiers the algorithm has three iterations. The first one considers only sole quasi-identifiers, the second iteration considers pairs and the third and last iteration uses the full set of quasi-identifiers. Nodes that are black have been checked for k-anonymity and proven non k-anonymous by looking up their frequency table, red node are k-anonymous either by inference from the generalization property or by checking the frequency table. Grey nodes have been inferred to be non k-anonymous by either the Generalization or Subset property. Looking at the lattice of the 3^{rd} iteration we notice that predictive tagging is greatly reducing the search space, only a fraction of the lattice has to be searched.

Incognito has the advantage of not using a particular metric. It does not depend on *Prec* nor *height*. It is also an algorithm that achieves k-anonymity faster than any other algorithm up to the date of publishing. Most importantly, the ideas behind Incognito have been used and upgraded to build state-of-the-art algorithms: OLA in 2009 [37] and Flash in 2012 [36] in term of hierarchy-based k-anonymity.

Optimization Lattice Anonymization (OLA) presented by El Eman et al in [37] is similar to Incognito. It directly starts with the full size lattice and utilizes wisely selected sublattices to predict the status of some nodes using the Generalization property. The sublattice are defined using a binary search. Since both algorithm output the same globally optimal solution the main criterion to rank them is using the number of computations or run time. [37] demonstrates that for most of the usual privacy datasets OLA checks up to four times less nodes than Incognito. On other datasets their performance are very similar.

The Flash algorithm introduced by F.Kohlmayer et al. in [36] is also using generalization lattice to establish the best

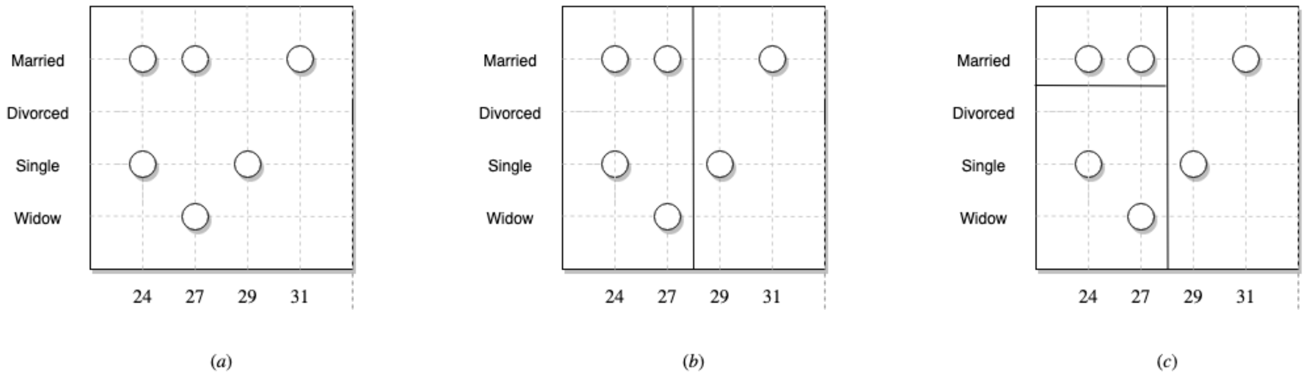


Fig. 6. (a) spatial representation, (b) first cut of the Mondrian algorithm, (c) second and last cut of the Mondrian algorithm, extracted from LeFevre et al. [16]

possible *k*-anonymous generalization of a table. Instead of relaying on sublattices, Flash builds path between unvisited nodes and the top using only unvisited nodes starting from the bottom using a depth-first search. Once a path is obtained it uses a binary search on the height to establish the status of each node of the path. Similarly to Incognito and OLA, Flash uses the three property to do some predictive tagging. Again using five of the most used datasets, the average execution times of Flash beats OLA and Incognito.

This section covered algorithms that use lattices. They are amongst the most popular algorithms [38] when it comes to practical anonymizations because hierarchies are a way guide the generalization process so that the data keeps its utility. Moreover the above algorithms are independent of metrics, indeed once the set of *k*-anonymous nodes is found we can use any metrics to pick up the best in each use case.

4.3 Mondrian

Presented by K.LeFevre, D.J.DeWitt and R.Ramakrishnan in 2006, Mondrian [16] is different from the previously presented algorithms. It is not using hierarchies and it is not a single dimension model: it is a global multi-dimension partition-based model.

A convenient data representation for partition-based model is the spatial representation. Each record of the table is represented by a dot in a N_{QI}-dimensional space. The representation of six records having two quasi-identifiers Age and Marital Status is displayed figure 6 (a).

The Mondrian algorithm's main idea is achieving a top-down clustering of the space by defining regions using *k*-dimensional trees [39] and greedily selecting each cut. This process is similar to how a Decision Tree classifier is constructed. A cut with a region is defined by an orthogonal dimension and a splitting point. At each iteration, the dimension with the widest normalized range of values is selected. This dimension corresponds to an attribute and the median value of this attribute is selected as splitting point.

All records in the same region are generalised to become the same record. Initially, Mondrian starts with all records in the same region, and then cuts the space divide a region in two smaller regions. In order to make sure that the result is giving clusters of at least *k* elements, LeFevre et al. defined *allowable cuts* [16]. A cut is allowable if both regions created by this cut have at least *k* elements. Only allowable cuts are executed so the result is guaranteed to be *k*-anonymous. The algorithm stops when there are no remaining allowable cuts.

Algorithm 3 Mondrian

```
Require: T table. $k$ fixed integer.
Ensure: $|T| > k$
  Anonymize(region):
  if no allowable cut in this region: then
    return region
  else
    $dim \leftarrow$ dimension with widest normalized range
    $med \leftarrow$ the median value for that attribute
    right $\leftarrow \{t \in$ region $: t[dim] > med\}|$
    left $\leftarrow \{t \in$ region $: t[dim] \leq med\}|$
  end if
  return Anonymize(left), Anonymize(right)
```

An example of Mondrian in action for 2-anonymity can be seen figure 6 (b) & (c). At the beginning both quasi-identifiers have a normalized range of values of 1. We arbitrary select one dimension: the Age. We browse for the median, it is bigger than 27 and smaller than 29 ; by cutting here the created regions have 4 and 2 data points so the cut is allowable. The first cut is orthogonal to the Age axis and between the value Age= 27 and Age= 29. For the second cut, we arbitrary consider the "left" region i.e smaller than the value point, here it is the biggest one in terms of data points. It has more than $2k = 4$ distinct points so it is worth looking for a cut. The normalized range of values is 1 for the Marital Status and 0.5 for the Age. We select the Marital Status and look for the median that is right before Divorced and after Single. The considered cut is allowable. The second cut is orthogonal to the Marital Status axis and between

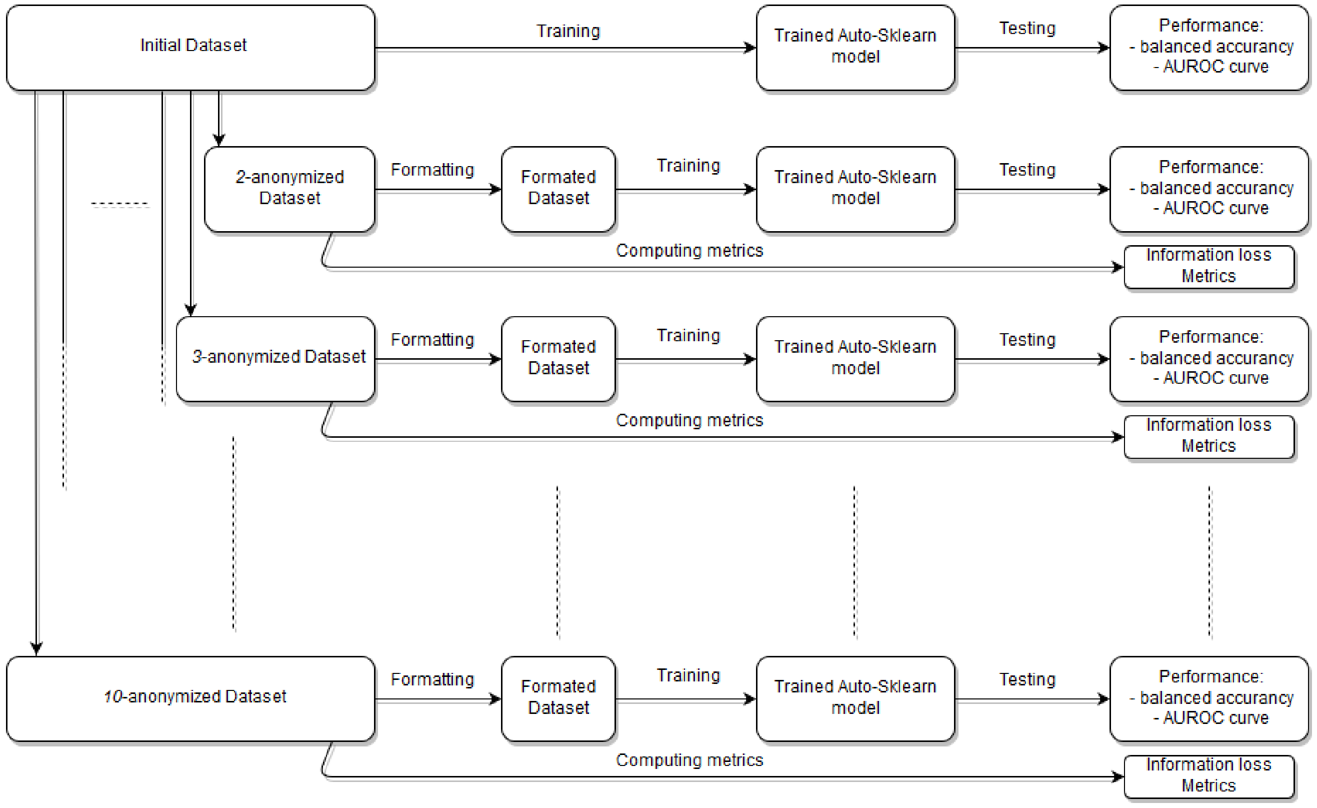


Fig. 7. The experimental methodology.

the values Single and Divorced. All the remaining regions only have $2 < 4$ data points so no allowable cuts are possible, the algorithm ends. We end up with 3 clusters each with at least $k = 2$ elements. Each point's values are generalized into the interval described by his region. For example, the point at the top right corner use to be (Married, 31) and it is generalized into ({Married, Divorced, Single, Widow}, {29, 31}). The algorithm 3 details the process [16].

Mondrian guarantees k-anonymity but not optimality with regards to a particular metric. According to [16] a multi-dimension approximate solution can offer better results than an optimal single-dimension solution. Thanks to the two heuristics to select the dimension and the splitting point, the computation time is reduced compared to previous optimal algorithms [16].

Table 5. Details regarding features of the credit dataset.

Number	Explanation	Type
1	Amount of credit given (integer)	QI
2	Gender ([2])	QI
3	Education ([3])	QI
4	Marital Status ([4])	QI
5	Age (integer)	QI
6-11	History of payment over the last 6 months	Sensitive
12-17	Amounts of bill statement over the last 6 months	Sensitive
18-23	Amounts of payment over the last 6 months	Sensitive
24	Default payment	Sensitive

5 Experiment and Results

Data is rarely an end but it is rather a mean to do a specific task. For example, data can be a mean to gain information through analytics like TfL did [4] or to train a machine learning model. As mentioned earlier in section 3 it is difficult to quantify the data utility with respect to a task after it has been anonymized. Current metrics enable a selection of the best generalization(s) amongst the set of all k-anonymous generalizations but they don't give insights regarding the impact of the anonymization on the task's performances.

To pinpoint this issue we are comparing current metrics with the difference of performance before and after anonymization on an actual machine learning task. The chosen task is a classification problem where given personal and historical financial data about someone we have to predict if this person is going to default on their credit card the following month. The dataset was used in 2009 by Yeh et al. in [40] and publicly released in 2016 to the UC Irvine Machine Learning Repository .

The already-pseudonymized dataset contains 30000 records and 24 attributes detailed in table 5. We consider

[2] (1=male; 2=female)

[3] (1=grad school; 2=undergrad; 3=high school; 4=others)

[4] (1=married; 2=single; 3=others)

features $1-5$ to be quasi-identifiers because together they can uniquely identify someone in the dataset and they are publicly available. One could argue that the amount of credit given is not publicly available but we make the assumption that most banks offer standard services and that an attacker can infer a rough estimate of the credit allowed to somebody. Those five quasi-identifiers are encoded as integers because they can be seen as ordered. The eighteen other features are considered sensitive and have to be protected. Given those 23 features, our goal is to predict the label that is the twenty-fourth attribute of this dataset.

5.1 Methodology

Of the algorithms we presented earlier, we chose to use Mondrian [16]. We used the publicly available implementation of Qiyuan Gong [41]. The main arguments for using Mondrian are that it does not depend on hierarchies so we don't have to define hierarchies that would narrow the generality of the results. Moreover, it is quick and easy to compute even with limited compute power e.g. on a laptop. The values are either left untouched or generalized into an interval. Intervals cannot be used by usual machine learning models which mostly take numerical values in input, so in order to make them readable by our model, we replaced each interval column with two columns. The two columns contain the left boundary and the right boundary of the intervals. For example, if the initial value of a record was Age$=22$ and it was generalized into the interval $[20-27]$, then the values encoded in the two new columns would be Age$_{\text{min}}=20$ and Age$_{\text{max}}=27$. The formatted dataset has 29 features compared to the 24 of the initial dataset because the five quasi-identifiers now account for ten columns.

The machine learning models that we use for classification that we use are auto-sklearn models [42]. Those models leverage bayesian-optimization to do automated algorithm selection within the Scikit-learn [43] python library [44] and hyper-parameters tuning. We chose auto-sklearn because the only hyper-parameter of the model is the training time, by giving the same training time to models training on different datasets we aim at comparing then of a fair ground. Each model was trained during one hour.

The experimentation methodology is as follows. From the initial credit dataset we compute its $2,3,4,5,6,7,8,9,10$-anonymizations with the Mondrian algorithm detailed above. We have the original datatset and nine anonymized datatsets. We compute the information loss metrics Π_e(represented by the letter H), NCP, DM and CM for each dataset. The nine anonymized datasets are then formatted so that intervals are replaced by their left and right boundaries to be valid input for our auto-sklearn models. We end up with one 30000×24 and nine 30000×29 tables. The ten datasets are divided in Train/Validation and Test with a ratio 90/10 i.e. 27000 records from training and validation and 3000 for testing. We train 10 auto-skearn models, one on each of the ten datasets. Using the test sets we compute the balanced accuracies and the Areas Under the Receiver Operating Characteristic(AUROCs) curves. Figure 7 sums up the methodology used. This methodology is applied three times and the results are averaged.

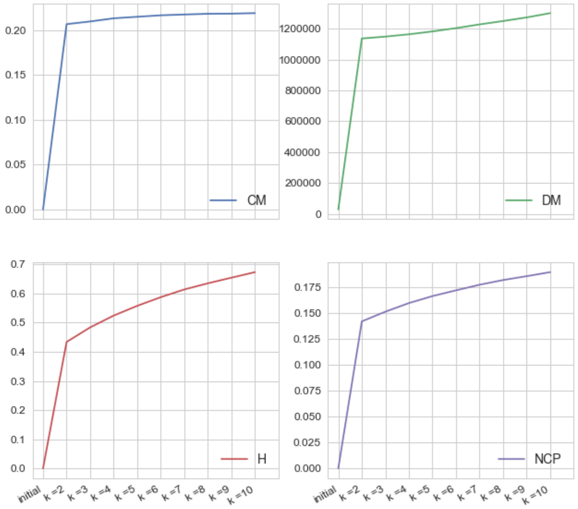


Fig. 8. Plots of Information Loss metrics with respect to each dataset

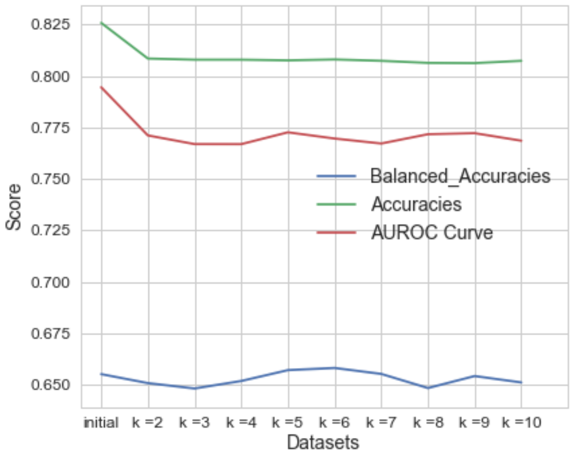


Fig. 9. Plots of the classification metrics with respect to each model

The machine used is a Dell XPS-15 under 64-bit Ubuntu 18.04.2 LTS with a Intel® Core$^{\text{TM}}$i7-8750H CPU @ 2.20GHz x 12 and 16 GiB of RAM.

5.2 Results:

First, we evaluate the anonymization with the metrics presented in section 3. Figure 8 displays the information loss metrics for the ten datatsets. Looking at the plots, all metrics are behaving in a similar fashion, resembling a a Heaviside step function. The 2-anonymization does most of the harm and further anonymization is not making the results much worse. This is especially true for CM and DM. This can be explained by the fact that CM and DM mainly look at the size of the equivalence classes while H and NCP take into account each cells of the generalized table.

The number of equivalence classes shows that they are

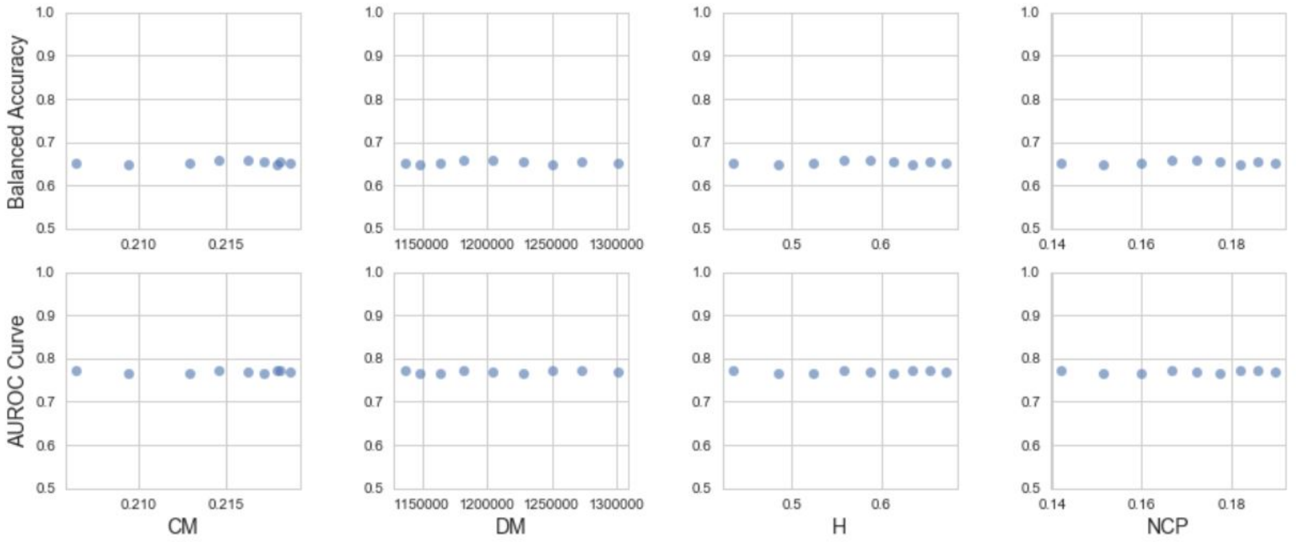


Fig. 10. Performance metrics plotted against the information loss metrics for the nine anonymized datasets

two to three times bigger than they should be. Some equivalence classe are really big and that explains the skyrocketing values of DM. This is probably specific to the dataset that is naturally compacted i.e. a few hundreds of records are very close too each other in the spatial representation. Entropy and NCP are slowly and steadily increasing after the 2-anonymization. Difficult to guess the real effect of those anonymizations on the data. The Classification Metric CM tells us that the average label ratio in a cluster is 80/20 but since quasi-identifiers make around a fifth of all the features it only gives us an idea of what are in those clusters and it cannot be turned into valuable utility insight.

The average performances of the ten Auto-sklearn models are displayed in figure 9. The accuracy is not very telling because the dataset is naturally imbalanced with approximately 78% of non-default and 22% of credit default. We are therefore more interested in the balanced accuracy(blue line) and the AUROC curve(red line). We see a small drop in the AUROC curve for the 2-anonymized dataset but after that the curve is flat. The balanced accuracy is not impacted by anonymization as a score around 0.650 is achieved for all ten classifiers.

Unexpectedly, each classifier seems to be performing at roughly the same level. The anonymization has little impact on the overall performance. As can be seen on figure 10, when plotting a classifier's metric against an information loss metric we only obtain horizontal patterns. Each dot is a classifier, its y-value is its performance and the x-value is the information loss metric of the dataset it is trained on. The huge peaks for all four information loss metric(visible in figure 8) is not followed by a drop of the balanced accuracy.

We have to keep in mind that only five attributes were anonymized among the twenty-three available. The five quasi-identifiers have been selected because of their high risk of re-identification but their predictive power is maybe not as important as financial attributes that we considered sensitive attributes.

It is difficult to see a real correlation between the metrics currently used to computation information loss while doing *k*-anonymization and the scores achieved by classifiers trained on the anonymized dataset. The metrics show a clear deterioration of the data while no consequences are witnessed on the performance of the classifiers.

6 Related Work

Disclosure control, the art of releasing data that offer no possibility of re-identification has been a popular field of research. Other anonymization methods exist such as adding noise to the dataset [45] [46] or modifying the dataset while keeping statistical invariants [47]. Those methods have the downside of losing the truthfulness of the data. After the introduction of *k*-anonymity by Sweeney [8], other anonymization criteria have been proposed to tighten the release of data and cover additional potential attacks. Among them, *l*-diversity imposes that at least *l* diverse values for sensitive attributes are in each equivalence class, protecting equivalence classes which might have uniform sensitive values. *t*-Closeness [48] further refines this idea by imposing constrains on the distribution of the sensible attributes in each equivalence classes. k^m-anonymity [49] makes a formal hypothesis that the attacker knows any set of *m* records from the dataset and ensures that those *m* records cannot be leveraged to identify fewer than *k* tuples in the dataset.

7 Conclusions and Future Work

In this paper, we presented the need for anonymization, then we offered a taxonomy of generalization models followed by a review of current information loss metrics and algorithms. This permitted to understand the landscape surrounding *k*-anonymization. The different loss metrics were used in a real case study to evaluate their ability to provide insights over the data utility in a machin learning task.

The models have been trained during one hour, this draws a limit to what can be concluded. The performance gap, currently nonexistent as shown in our results may appear if more complex models are considered. The reproduction of the above work at greater scale and on a greater time frame can strengthen the obtained results, indeed using a wider variety of datasets and tasks is yet to be done. Future work may include the introduction of a benchmark constructed with popular datasets and their respective tasks.

Acknowledgements

I would like to thank Florimond Houssiau for supervising and supporting me in this project especially for his countless feedbacks. I am also grateful to Yves-Alexandre de Montjoye and Imperial College for the opportunity they gave me working on this project.

References

[1] Internet World Stats. Internet Growth statistics;. [Accessed 3rd April 2019]. [Online]. Available from: https://www.internetworldstats.com/emarketing.htm.

[2] Goldman Sachs Global Investment Research . The Internet of Things: Making sense of the next mega-trend;. [Accessed 3rd April 2019]. [Online]. Available from: https://www.goldmansachs.com/insights/pages/internet-of-things/iot-report.pdf.

[3] The Economist. Data, data everywhere;. [Accessed 3rd April 2019]. [Online]. Available from: https://www.economist.com/special-report/2010/02/27/data-data-everywhere.

[4] Tfl. Review of the TfL WiFi pilot: our findings;. [Accessed 17th April 2019]. [Online]. Available from: http://content.tfl.gov.uk/review-tfl-wifi-pilot.pdf.

[5] Sky News. Tfl plans to make 322m by collecting data from passengers' mobiles via Tube Wi-Fi;. [Accessed 17th April 2019]. [Online]. Available from: https://news.sky.com/story/tfl-may-make-322m-by-selling-on-data-from-passengers-mobiles-via-tube-wifi-11056118.

[6] Space Machine. Datasets Over Algorithms;. [Accessed 3rd April 2019]. [Online]. Available from: http://www.spacemachine.net/views/2016/3/datasets-over-algorithms.

[7] McKinsey Global Institute. The age of analytics: Competing in a data-driven world;. [Accessed 17th April 2019]. [Online]. Available from: https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/the-age-of-analytics-competing-in-a-data-driven-world.

[8] Sweeney L. K-anonymity: A Model for Protecting Privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 2002 Oct;10(5):557–570.

[9] de Montjoye YA, Hidalgo CA, Verleysen M, Blondel VD. Unique in the Crowd: The privacy bounds of human mobility. In: Scientific reports; 2013. .

[10] de Montjoye YA, Radaelli L, Singh VK, Pentland A. Unique in the shopping mall: On the reidentifiability of credit card metadata. Science. 2015;347(6221):536–539.

[11] Sweeney L. Achieving K-anonymity Privacy Protection Using Generalization and Suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 2002 Oct;10(5):571–588.

[12] Bayardo RJ, Agrawal R. Data privacy through optimal k-anonymization. In: 21st International conference on data engineering (ICDE'05). IEEE; 2005. p. 217–228.

[13] Gionis A, Tassa T. k-Anonymization with minimal loss of information. IEEE Transactions on Knowledge and Data Engineering. 2009;21(2):206–219.

[14] Dalenius T. Finding a needle in a haystack or Identifying anonymous census records. Journal of Official Statistics. 1986;2:329–336.

[15] Samarati P, Sweeney L. Protecting Privacy when Disclosing Information: k-Anonymity and its Enforcement through Generalization and Suppression; 1998.

[16] LeFevre K, DeWitt DJ, Ramakrishnan R, et al. Mondrian multidimensional k-anonymity. In: ICDE. vol. 6; 2006. p. 25.

[17] Ayala-Rivera V, McDonagh P, Cerqueus T, Murphy L. A Systematic Comparison and Evaluation of k-Anonymization Algorithms for Practitioners. Trans Data Privacy. 2014;7(3).

[18] Ciriani V, di Vimercati SDC, Foresti S, Samarati P. k -Anonymous Data Mining: A Survey. In: Privacy-Preserving Data Mining; 2008. .

[19] LeFevre K, DeWitt DJ, Ramakrishnan R. Incognito: Efficient full-domain k-anonymity. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data. ACM; 2005. p. 49–60.

[20] De Waal A, Willenborg L. Information loss through global recoding and local suppression. Netherlands Official Statistics. 1999;14:17–20.

[21] Aggarwal G, Feder T, Kenthapadi K, Motwani R, Panigrahy R, Thomas D, et al. Anonymizing tables. In: International Conference on Database Theory. Springer; 2005. p. 246–258.

[22] He Y, Naughton JF. Anonymization of Set-valued Data via Top-down, Local Generalization. Proc VLDB Endow. 2009 Aug;2(1):934–945.

[23] Meyerson A, Williams R. On the complexity of optimal k-anonymity. In: Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. ACM; 2004. p. 223–228.

[24] Xu J, Wang W, Pei J, Wang X, Shi B, Fu AWC. Utility-based anonymization using local recoding. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2006. p. 785–790.

[25] Ayala-Rivera V, McDonagh P, Cerqueus T, Murphy L. A Systematic Comparison and Evaluation of k-Anonymization Algorithms for Practitioners. Trans

Data Privacy. 2014 Dec;7(3):337–370.

[26] Ghinita G, Karras P, Kalnis P, Mamoulis N. Fast data anonymization with low information loss. In: Proceedings of the 33rd international conference on Very large data bases. VLDB Endowment; 2007. p. 758–769.

[27] Iyengar VS. Transforming data to satisfy privacy constraints. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2002. p. 279–288.

[28] Willenborg L, de Waal T. Elements of Statistical Disclosure Control. Lecture Notes in Statistics. Springer New York; 2012.

[29] Cover TM, Thomas JA. Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing). New York, NY, USA: Wiley-Interscience; 2006.

[30] Shannon CE. A mathematical theory of communication. Bell system technical journal. 1948;27(3):379–423.

[31] Sweeney L. Datafly: A System for Providing Anonymity in Medical Data. In: Proceedings of the IFIP TC11 WG11.3 Eleventh International Conference on Database Securty XI: Status and Prospects. Chapman & Hall, Ltd.; 1998. p. 356–381.

[32] Sweeney L. Computational Disclosure Control for Medical Microdata. 1997 1;.

[33] Sweeney L. Guaranteeing anonymity when sharing medical data, the Datafly System. Proceedings : a conference of the American Medical Informatics Association AMIA Fall Symposium. 1997;p. 51–5.

[34] Samarati P. Protecting respondents identities in microdata release. IEEE transactions on Knowledge and Data Engineering. 2001;13(6):1010–1027.

[35] Wang K, Philip SY, Chakraborty S. Bottom-Up Generalization: A Data Mining Solution to Privacy Protection. In: ICDM. vol. 4; 2004. p. 249–256.

[36] Kohlmayer F, Prasser F, Eckert C, Kemper A, Kuhn KA. Flash: efficient, stable and optimal k-anonymity. In: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing. IEEE; 2012. p. 708–717.

[37] El Emam K, Dankar FK, Issa R, Jonker E, Amyot D, Cogo E, et al. A globally optimal k-anonymity method for the de-identification of health data. Journal of the American Medical Informatics Association. 2009;16(5):670–682.

[38] Terrovitis M, Tsitsigkos D. Amnesia;. [Accessed 3rd April 2019]. [Online]. Available from: https://amnesia.openaire.eu/contact.html.

[39] Friedman JH, Bentley JL, Finkel RA. An Algorithm for Finding Best Matches in Logarithmic Expected Time. ACM Trans Math Softw. 1977 Sep;3(3):209–226.

[40] Yeh IC, hui Lien C. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Syst Appl. 2009;36:2473–2480.

[41] Qiyuan Gong. Python Implementation for Mondrian Multidimensional K-Anonymity (Mondrian);. [Accessed 9th February 2019]. [Online]. Available from: https://github.com/qiyuangong/Mondrian.

[42] Feurer M, Klein A, Eggensperger K, Springenberg J, Blum M, Hutter F. Efficient and Robust Automated Machine Learning. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R, editors. Advances in Neural Information Processing Systems 28. Curran Associates, Inc.; 2015. p. 2962–2970.

[43] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12:2825–2830.

[44] Buitinck L, Louppe G, Blondel M, Pedregosa F, Mueller A, Grisel O, et al. API design for machine learning software: experiences from the scikit-learn project. In: ECML PKDD Workshop: Languages for Data Mining and Machine Learning; 2013. p. 108–122.

[45] Kim J. A method for limiting disclosure in microdata based on random noise and transformation. 1986;p. 370–374.

[46] Mivule K. Utilizing noise addition for data privacy, an overview. arXiv preprint arXiv:13093958. 2013;.

[47] Duncan GT, Pearson RW, et al. Enhancing access to microdata while protecting confidentiality: Prospects for the future. Statistical Science. 1991;6(3):219–232.

[48] Li N, Li T, Venkatasubramanian S. t-closeness: Privacy beyond k-anonymity and l-diversity. In: 2007 IEEE 23rd International Conference on Data Engineering. IEEE; 2007. p. 106–115.

[49] Terrovitis M, Mamoulis N, Kalnis P. Local and global recoding methods for anonymizing set-valued data. The VLDB JournalThe International Journal on Very Large Data Bases. 2011;20(1):83–106.