De-identification of clinical time series for predictive modeling

Leonardo Mazzone

MInf Project (Part 2) Report

Master of Informatics
School of Informatics
University of Edinburgh

2020

Abstract

The increasing availability of digitized health data to governments and researchers unlocks unprecedented opportunities for improving healthcare and advancing the medical professions through data-intensive applications. However, these opportunities could become threats to the privacy of individuals when large datasets are released without the appropriate protections. It is therefore necessary to apply privacy-conscious methods and de-identify data. The most widely applied de-identification definition is k-anonymity, enforced through generalization and suppression of data. However, that is not applicable to time series databases, because of the *curse of dimensionality*. On the other hand, time series are the most natural way of recording clinical information powering beneficial applications. This project attempts to bridge the gap by introducing a new privacy metric and a related anonymization technique for time series. These rely on the assumption that malicious actors can exploit known time series features to breach privacy. This work also demonstrates the application of these methods to MIMIC-III, a large medical dataset, and discusses their effectiveness in practice, additional adaptations, and the resulting data degradation in terms of the performance of an illustrative data mining task.

Acknowledgments

I would like to thank Dr. Markulf Kohlweiss for his continued friendly support through these past two years, his ever-insightful questions and observations, and for making almost all of our meetings.

Thanks to Dr. Korin Reid and Craneware for their interest in this work and for all the useful chats about their experience with the healthcare industry.

This project was completed at a rather distressing time for me, the university community and the world at large. A huge thanks to all the people that in the face of difficulties allowed me to finish the work smoothly. This includes the invaluable logistical and emotional support from my family, Rachel, Lukasz and Ignat.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Summary of contributions . . . 2
- 1.3 Report structure . . . 3

2 Background **5**
- 2.1 Health informatics in critical care . . . 5
- 2.2 Data de-identification . . . 7
- 2.3 Previous work . . . 9
- 2.4 Time series anonymization . . . 11

3 Theoretical framework **15**
- 3.1 Problem definition . . . 15
- 3.2 Feature k-anonymity . . . 16
- 3.3 Feature-similarity disagreement . . . 18

4 Algorithmic implementation **23**
- 4.1 k-anonymizing features . . . 23
- 4.2 Constraint optimization . . . 26
- 4.3 Computing disagreement . . . 28

5 Application to MIMIC-III **31**
- 5.1 Identifying features . . . 31
- 5.2 Full-pipeline overview . . . 32
- 5.3 Time series generation and pre-processing . . . 33
- 5.4 Duration de-identification . . . 34
- 5.5 Feature de-identification . . . 35
- 5.6 Privacy evaluation . . . 37
- 5.7 Utility evaluation . . . 38

6 Results **43**
- 6.1 De-identifying lengths of stay . . . 43
- 6.2 Tuning the constraint optimization . . . 44
- 6.3 Feature de-identification . . . 46
- 6.4 Trust settings and model performance . . . 50

7 Conclusion **53**

7.1 Discussion of experiments 53
7.2 Future work 54

Bibliography **55**

A Summary of notation **61**

B Data management plan **63**

Chapter 1

Introduction

1.1 Motivation

The fields of artificial intelligence and data mining are developing at an impressive pace due to a dynamic and well-funded research community in both the public and private sectors, and to the increasing availability of rich datasets. Better automated data analysis and prediction could prove "inestimable for virtually every industry and field of scientific research, allowing to spot patterns, build prediction systems, and optimize processes and resources" [29]. The medical field is not an exception, thanks both to advancements in deep-learning models, and to the electronic recording of patients data, motivated at first by administrative and financial necessities, and today becoming the standard in many developed countries [41]. In fact, research in health informatics promises to greatly improve healthcare by cutting costs, identifying most urgent cases, and improving diagnosing.

However, as scandals accumulate (see for example [33]) and the general awareness increases of the privacy issues inevitably tied to large-scale data collection, it is urgent to re-evaluate data management practices. Indeed, it is now evident that wide availability of data is both central to the development of modern AI, and exploitable to learn private information, with negative effects both at the individual and societal level. Once again, this is extremely relevant to healthcare, a context in which the data collected tends to be extremely sensitive.

While researchers have been prolific in the production of theoretical definitions of privacy and anonymization algorithms, instances of the application of these results are dishearteningly scarce. The problem is in part cultural, or due to lack of incentives. However, the truth is that to this day effective anonymization remains a hard goal to achieve, both in terms of privacy protection and of the preservation of data integrity. The quest for data anonymity is arduous for at least the following reasons:

- It is exponentially harder to remain anonymous in the face of the wealth of data of today's society. Individuals can be uniquely identified by collections of properties that would intuitively (and wrongly) be thought to be harmless. The combination of information from different data sources can be exploited by

uncovering relations between people and truly-sensitive data attributes, with real and potentially serious life consequences.

- It is almost impossible to trace a line between data that is supportive to legitimate instead of malicious purposes. This line mostly lies in the intentions of the data recipient. Therefore, anonymization and preservation of data utility are inherently antithetic goals and what should be sought is an appropriate trade-off. At the same time, limiting access to data is unappealing because it can stifle scientific development and innovation. Decreasing the quality of the data appears then unavoidable.

- As common to the field of cybersecurity, absolute protection is but a chimera. Instead, it is necessary to build a threat model and develop strategies that allow to minimize risks.

All the above points lead to the conclusion that there exists no one-size-fits-all solution to the problem of data privacy. Appropriate assessments need to be carried out such that context-specific solutions can be deployed taking multiple factors into account, including the gravity and potential impact of privacy breaches, the most likely adversaries, and the costs in terms of the missed opportunities of better data-intensive applications (which might be quantified in Euros or in human lives). This paper attempts to find a useful trade-off in the under-studied setting of clinical time series, particularly those produced in the Intensive Care Unit, where the main technical difficulty is the extremely high-dimensionality of data, and one of the most concerning privacy threats is the diffusion of consumer health-trackers.

1.2 Summary of contributions

- Extensive literature review of existing approaches to time series anonymization and discussion of their suitability to healthcare data mining applications.

- Identification of privacy concerns related to health measurements in MIMIC-III, a publicly-available dataset of Electronic Health Records, in light of the diffusion of health-tracking devices.

- Formulation of feature k-anonymity, a generalization of k-anonymity applicable to high-dimensional data and time series.

- Formulation of feature-similarity disagreement, a novel privacy measure that permits the analysis of a large class of sensitive value estimation attacks.

- Design of a de-identification algorithm for time series based on the Mondrian algorithm and on optimization, whose goal is to increase the level of privacy protection as measured by the metrics introduced in this project.

- Implementation of a data transformation pipeline that allows the application of the de-identification algorithm on MIMIC-III, optimizes it, and permits its systematic evaluation.

- Experimental assessment on the MIMIC-III dataset of the de-identification algorithm with respect to the resulting privacy gain. Evaluation of the remaining utility in the de-identified data based on the performance of a benchmark model trained on the anonymized data to predict the in-hospital mortality of patients.

1.3 Report structure

Chapter 2 introduces the importance and difficulty of anonymizing clinical time series, by outlining some of the current research in health informatics, summarizing classical privacy definitions in anonymization literature, and more recent attempts at anonymizing time series.

Chapter 3 gives a rigorous definition to the problem this project addresses, introduces notation, and formulates a theoretical anonymization framework that extends and generalizes the ideas of k-anonymity and l-diversity such that they can be applied to time series in practice.

Chapter 4 describes the algorithmic implementation of the anonymization framework.

Chapter 5 deals with the instantiation of the anonymization framework such that it can be applied to MIMIC-III. In doing so it discusses the pre-processing of the dataset, adaptations of previously discussed algorithms and the machine learning model that was used for the purpose of evaluation.

Chapter 6 motivates, describes, and presents the outcome of a set of experiments on the MIMIC- III dataset, with the goal of analyzing privacy vulnerabilities of time series, assessing the privacy gain granted by the anonymization techniques developed, and evaluating their interaction with a benchmark data mining task (in-hospital mortality prediction).

Chapter 7 discusses the results of previous experiments drawing conclusions, provides suggestions for future work, and summarizes the accomplishments of the project.

Chapter 2

Background

2.1 Health informatics in critical care

Providing a brief outline of the main research directions and opportunities in health informatics allows us to gain some insight into the specificities of the data needs in this domain. While this section does not have the ambition of touching upon every aspect of this rich field, noteworthy examples are reported in order to address some important trends, in particular those pertaining to prediction and automated decision-making for critical care.

"Precision medicine" is a medical model that is gaining traction, and refers to the exploitation of individual characteristics (in gene, environment, and lifestyle) to produce customized practices and treatment and thus improve outcomes and prevent the onset of conditions [28]. Unsurprisingly, several machine learning techniques lend themselves to the automation of these tasks. The quick training (in comparison to human learning), high availability and consistent performance can make them great decision-support tools, especially in cases in which well-trained doctors are a scarce resource. Genomics might be a useful discipline for the study of personalized treatment, but today "most tests based on genomics are still too slow to be useful in the Intensive Care Unit, and the data they generate do not readily inform clinical decision-making" [28]. For this reason, most models rely on Electronic Health Records (EHR), or systematic digital databases of patients data, both in the form of demographics and histories of observation and treatment. EHR adoption has exploded in the last decade. In the USA, for instance, they have become ubiquitous for the facilitation of billing [41], but are a godsend to medical AI.

Sepsis, a condition that arises when a response to infection causes injury to the body's own tissues and organs, is the main cause of mortality in hospitals worldwide [13]. In [20], the authors train, using EHRs, an agent aimed at the identification of an optimal sequence of procedures for patients affected by sepsis. The work in [35] attempts to automatically determine good policies for mechanical ventilation (the usage of artificial means to assist spontaneous breathing), including a personalized regime of sedation dosage and time-to-extubation readiness. Other instances of AI-based tools for precision medicine include systems that establish appropriate administration rates

of insulin for diabetic patients, or of sequences of drugs in HIV therapy or cancer treatment [29].

Another important task is that of prediction: the system developed in [7] uses clinical histories for early detection of the development of sepsis in patients, making timely interventions possible. In another study, an innovative time-aware neural model has been employed to group heterogeneous patients into disease characterizing subtypes, or clusters defined by distinct progression patterns of the same disease, requiring different types of therapeutic intervention.

All the studies cited so far have some common features that must be emphasized. Firstly, they inevitably depend on sequential (or longitudinal) data. Be it the progression of discrete items (e.g. diagnosis codes from a standard codebook) or numerical values (vital signs), time is an inherently important dimension of clinical information. Secondly, the sparseness and irregularity of measurements demands resampling to constant time intervals as a pre-processing step, imputing missing values by means of interpolation. This makes the data easier to handle for most machine learning architectures, but I argue that it is also beneficial from the point of view of de-identification. Finally, most studies need to gather relevant cohorts of patients from the data through filtering techniques. Failure to do so accurately condemns the study to invalidity and tools to systematic errors. On one hand this highlights the importance of recording and receiving patient demographics. On the other hand, the most complex queries (e.g. "patients with diabetes", or "patients who had a recent heart failure") need data with sufficient granularity and dimensionality, to accommodate for generic needs that might not have been obvious at the time of data recording or anonymization.

The Laboratory for Computational Physiology at Massachusetts Institute of Technology released MIMIC-III [18], a database comprising EHR of patients admitted to the Intensive Care Unit of a large US hospital[1]. This release wants to help address the problem of reproducibility in biomedical research [5]. As a consequence, the data is distributed in a semi-open fashion[2], thus "allowing clinical studies to be reproduced and improved in ways that would not otherwise be possible". Until recently however, progress in the application of machine learning to healthcare has been difficult to demonstrate, due to the absence of publicly available benchmarks. The important contribution of [16] has been to propose four standard benchmarks against which to compare the merit of different models. The benchmarks rely on the MIMIC-III data, to maximize ease of access, and use some strong baselines, most notably Long-Short Term Memory (LSTM) deep neural networks. The goals of each benchmark include the following:

- Forecasting length of stay: not only is this important for scheduling and hospital resource management; conversations with Craneware[3] suggest that this is an important metric used in practice for the comparison of physician efficiency.

[1]The Beth Israel Deaconess Medical Center in Boston, Massachusetts

[2]As the data is quite sensitive, access requires approval from the data owners and the completion of an ethical training course

[3]Craneware is a prominent US company developing software that enables healthcare providers to improve margins and enhance patient outcomes, e.g. software to analyze insurance claims and predict costs. They have supported this project by providing useful insights and feedback.

- Modeling risk of mortality: useful for prevention and resource allocation.
- Detecting physiologic decline: another important early-warning predictor, signaling a likely rapid deterioration of the patient's health.
- Phenotype classification: this is closely related to automatic diagnosing. For each patient, the (multi-class classification) task is to identify the presence of any condition from a set of 25 common options in acute care.

In [44] the authors demonstrate with a newly-designed network that attention mechanisms achieve state-of-the-art performance in all the benchmark tasks, consistently beating LSTM sequential networks.

2.2 Data de-identification

Privacy legislation in the European Union and North America, particularly that pertaining to health data, often prescribes the removal of attributes from a defined set of known identifiers as a necessary and sufficient condition to legal compliance. In the USA for instance, this includes, under the "Safe Harbor" method, social security numbers, IP addresses, full-face photographs, precise geographical information and dates such as birthdates [29]. While the law recognizes that in some cases this is insufficient, devising compliant anonymization strategies is often up to the determination of experts, given the lack of recognition for more sophisticated methods as standards.

The reason the simple removal of direct identifiers is insufficient is that several, individually uninformative attributes can compound and become a uniquely identifying "signature" for an individual. Each of these is often called a *quasi-identifier*. Collections of quasi-identifiers can be exploited across different data sources (or using any background knowledge) to link individuals in a database that received shallow anonymization to sensitive attributes. This was for instance the case in the "Netflix incident" [33], where users were identified by matching their movie ratings to those available on the website IMDB. In 1997, Latanya Sweeney, the director of the Data Privacy Lab at Harvard, was able to re-identify the medical records of the then governor of Massachusetts [46] by mapping quasi-identifiers in a "de-identified" dataset to a publicly-available electoral registry. Similar attacks can be conducted using newspaper stories (see Figure 2.1) [47].

The vast majority of past de-identification research has focused on time-invariant, *snapshot-based* privacy. These terms refer to the setting in which data is encoded in a traditional database table, which is fundamentally a 2-dimensional grid where the row axis represents distinct data samples and the column axis represents dimensions. In particular, most of the theory and methods developed revolve around the concept of k-anonymity [46] and its extensions.

k-anonymity needs a binary classification of columns into quasi-identifiers and *sensitive fields*. According to the model in its original formulation sensitive fields are completely secret and unobtainable by adversaries, and are indeed the values the model wants to protect. On the contrary, quasi-identifiers are considered part of adversarial knowledge.

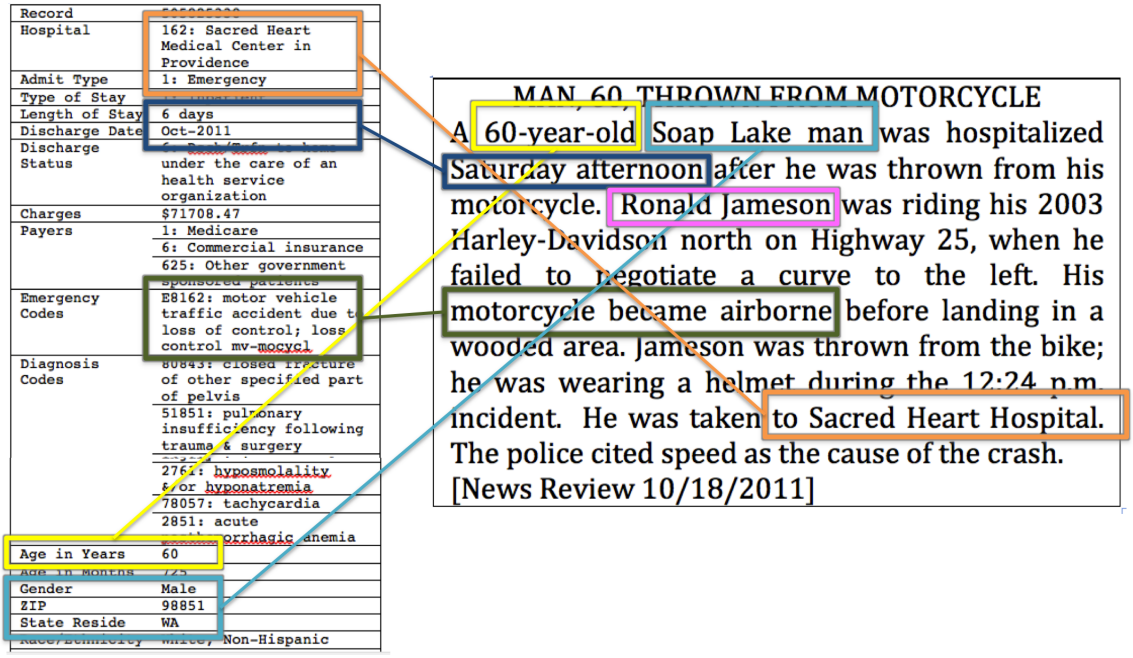

Figure 2.1: Sample re-identification with newspaper stories about hospital visit in Washington State [47]

This means that an attacker is able to map identities to the associated quasi-identifier values and could then obtain sensitive values.

Define *equivalence classes* as sets of all the records in a database that feature the same quasi-identifier values. A data release satisfies k-anonymity if all equivalence classes have size at least k. In this way records are protected by being "hidden in a crowd" of at least other k individuals. This definition is intuitive, but not particularly rigorous. However, k-anonymity has been shown under some conditions to provide formal guarantees of privacy, in particular by being exploitable to obtain the cryptographically-sound *differential privacy* model [26].

Datasets are normally made k-anonymous using the following two techniques:

- **Generalization**: different equivalence classes are unified by substituting their quasi-identifier values with new ones encompassing all of the previous classes. For example, adjacent, real numbers can be turned into ranges, adjacent ranges can be merged, and categorical values can be replaced with more generic categories.
- **Suppression**: records belonging to small equivalence classes are removed altogether.

Countless modifications of k-anonymity have been proposed to address its shortcomings. One issue is that the rigid distinction of values into identifying and secret appears difficult to justify and frail in hindsight, especially when new uncoordinated data releases happen after anonymization. Another problem is that k-anonymity does not capture at all the extent to which attackers can increase their knowledge on database participants. *Homogeneity attacks* are a well-known challenge that is possible when

in some equivalence classes all sensitive values are the same (or very similar), which neutralizes the protection of a crowd, regardless of its dimensions. l-diversity [27] and t-closeness [25] are two derivatives of k-anonymity that were formulated with homogeneity attacks in mind and will be dealt with in later chapters.

Arguably, however, the greatest problem of k-anonymity is that it cannot applied to high dimensional collections of quasi-identifiers. In the field of information privacy, this problem is known as the *curse of dimensionality*, and it can be intuitively illustrated as follows: since the number of combinations of possible quasi-identifier values grows exponentially with the number of attributes, so does the probability that within a population, a given combination is unique. More formally, effective k-anonymity is highly dependent on spatial locality, and spatial locality is no longer obtainable in high-dimensional spaces, where data becomes sparse [1]. For some types of data this problem is particularly severe. For instance, "anonymized" geographical data becomes identifying as soon as enough data points are available. If being logged in a location with a granularity of one square kilometer is usually not cause for concern, it takes only a few extra locations to make sure that the probability that other individuals share the same sequence of geolocation tags is negligible [42].

Differential privacy [8], briefly mentioned above, is a more recent and completely different approach, as it is not based on de-identification. It does not require any prior assumption on background knowledge as long as the records in the database are statistically independent. Differential privacy bounds the probability that a private output differs when removing any record from the original data, thus effectively limiting the consequences of the participation of individuals in a database. Differential privacy is attracting a lot of interest, but will not be the object of investigation of this report, for two reasons. Firstly, while being an attractive model in theory, it has some grave limitations, particularly with respect to data utility. It tends to be achieved with the addition of random noise on records. The noise falsifies the data in such a way that data consumers are only able to obtain accurate answers to a fixed, limited number of queries. Alternatively, only queries belonging to pre-specified classes can be performed. Research on how to use differentially-private datasets as input to generic neural architectures is lacking. Often the data is not even released, but interrogated through an interface. Secondly, my objective is to argue that traditional de-identification techniques and ideas, however imperfect, can go a long way in the protection of privacy, given appropriate adaptations.

2.3 Previous work

The first part of this project [29] focused on the development of anonymization techniques for MIMIC-III in the context of snapshot-based privacy. In other words, the focus was on protecting the privacy of patients with respect to demographics in tabular form. Some potential privacy risks linked to small equivalence classes in the Patients Admission table were exposed. These left the table vulnerable to linkage attacks.

A k-anonymizing algorithm called Optimal Lattice Anonymization (OLA) [9] was initially applied while considering several demographics as quasi-identifiers. The Adult

dataset [2] was also anonymized with the same procedure and the accuracy of binary income classification was used as an extrinsic measure of retained utility, showing very encouraging results (a small drop in accuracy). Subsequently, admissions data from MIMIC-III was joined with short sequences of procedures (referenced by a standard clinical identifier). In this setting OLA was shown to result in insufficient data utility, even though it outputs globally-optimal solutions, with respect to a utility function and given parameters k and a maximum tolerable suppression [4].

To address this issue, two distinct approaches were introduced. An algorithm called 'inverse OLA' was designed, reversing the optimization problem by looking for the best achievable value of k, given a maximum tolerable information loss. Additionally, a generalization of k-anonymity, named m-concealing, was formulated. This leverages a probabilistic model of an attacker's background knowledge. Each parameter of the model describes the probability of having full knowledge of the mapping between individuals and an attribute whose values are represented in a column of the database table we aim to protect. Using m-concealing as a privacy metric to optimize allowed the relaxation the requirements of k-anonymity, while considering more columns as possible quasi-identifiers, thus greatly increasing both the output utility and the confidence in the privacy protection mechanism.

Another important contribution was the development of an algorithm called ϵ-safe Lattice Anonymization, which combined the guarantees of k-anonymity and of differential privacy, at the cost of an output with larger information loss. The algorithm, ϵ-safe LA, was based on the non-deterministic sampling of generalization strategies from options represented as nodes in a lattice, with a probability distribution proportional to the utility of each node. In order to provide its theoretical guarantees, all attributes must be considered quasi-identifiers.

The greatest shortcoming of the results achieved in the first part of the project was their poor resilience to the curse of dimensionality. Even though it was possible to extend de-identification techniques to preserve short sequences of procedure codes, the methods previously analyzed do not scale well with growing sequences. It needs to be appreciated that complete sequences of discrete-valued events and vital measurements, i.e. time series, are the most interesting component of MIMIC-III and in fact any collection of Electronic Health Records. They are key to the sort of models introduced in Section 2.1 and to most useful applications. Even without seeking to achieve differential privacy, long time series are impossible to de-identify without devastating loss, because highly-dimensional data becomes unique, and thus identifying, much faster than linearly in the number of dimensions. Hence, new approaches need to be devised to handle this type of data in MIMIC-III.

[4] The solutions of OLA are globally optimal in the context of global, single-dimensional recordings [50]. This type of recordings restricts the space of feasible generalizations but assures a degree of consistency which was deemed important in the data analysis scenarios considered.

2.4 Time series anonymization

In this section I present an outline of several techniques that have been devised to anonymize time series. Some of them are based on ideas that inspired this project, others are mentioned to exemplify the limitations of current results, especially with the application to the health domain in mind. While time series data is ubiquitous in countless areas, from finance to climatology, two trends have recently made time series privacy research gain momentum. First of all, the popularity of geotagging: it is nowadays overwhelmingly common for individuals to be constantly equipped with GPS tracking devices, and intentionally or implicitly sending their coordinates to a third-party. The data they generate can have useful applications like urban planning but, as mentioned before, can also have severe privacy implications, causing the emergence of targeted solutions for location sequence anonymization (see for example [17, 12]).

The second phenomenon is the diffusion of the smart grid in several developed countries. The term "smart grid" refers to the infrastructure and appliances that facilitate the monitoring of energy consumption and its distribution across the network. This allows to more intelligently balance demand and supply with undeniable efficiency gains. Unfortunately, several attacks have emerged exploiting smart grid measurements. For example, in [32] the authors use power consumption patterns to infer information such as sleeping and eating routines and how many people are in a house at any time, while [14] succeeds in decoding what television program is being watched. Studies in smart grid data anonymity tend to take advantage of the fact that both the useful characteristics in this type of series (e.g. correctness of aggregate consumption across several households), and the characteristics that put privacy at risk (e.g. lack of smoothing in series pertaining to the same household) are understood reasonably well. The very interesting approach of [22] defines both utility and privacy requirements (that can vary across series) as constraints to satisfy by solving an optimization problem. This framework is flexible enough that it can formulate differential privacy as one of the constraints to satisfy. Results are demonstrated using some of the same utility and privacy criteria characteristic of the smart grid setting that I described. In [19], Pufferfish, a generalization of differential privacy that defines arbitrary secrets to maintain, is applied. Regrettably, there does not seem to be, to date, any comprehensive study regarding attacks on clinical time series, and thus the appropriate instantiation of these frameworks is uncertain.

[52] addresses the problem of time series processed on cloud computing infrastructures, and service request logs are proposed as an example of this issue. This study suggests to create fictitious items and ingest them in the data pipelines so that malicious service providers cannot tell fake requests from legitimate ones. This approach does not seem relevant to medical data mining since creating fake data can have serious consequences on the accuracy of models. On the other hand, [51] advocates for the introduction of synthetic observations in clinical time series (so as to protect rare patterns in consecutive records through k-anonymity). This however only applies to n-gram models, since the output is an anonymous matrix of n-gram frequencies.

Turning to methods based on the addition of noise to records, several papers recognize that the noise cannot be sampled independently from the same distribution. This

is because the correlations along the time dimension can be leveraged and the noise filtered using standard time series analysis techniques, like Fourier Transforms. A method called CTS-DP [49] counteracts this and obtains differential privacy by making noise dependent and computed on the basis of the autocorrelation of the original series. Because of the dependance on the autocorrelation function, series handled through this strategy must be stationary, that is, the statistical properties of the generating process do not change over time. This automatically excludes medical observations from the ICU. Similarly [34] develop a method that strikes a balance, in the computation of noise, between randomness (so that noise cannot be inferred when some value is leaked) and determinism (that avoids filtering attacks). However, privacy is simply measured as the standard deviation of the difference between true and released (perturbed) series. Another paper [54] acknowledging separation techniques from statistical signal processing, only deals with anonymization in which the output is a signal made of aggregated series. In our setting we care about accessing the individual series directly.

Sensitive characteristics of time series (amplitude, average, peak and trough, trend and periodicity) are listed in [55]. The paper continues by arguing that because of the number and diversity of sensitive characteristics, a generic privacy leakage measure is needed, and correlation between original and perturbed series is proposed. However, no indication on the concrete privacy protection offered by different correlation values is offered. The suggested anonymization strategy is the lowering of the resolution of records through discretization. Instead of generalization of data points, in [38] the authors propose to remove data points by selecting the least informative ones with a greedy strategy, but they do not suggest a way to quantify the resulting privacy gain.

In [43], authors suggest breaking time series into a twofold representation, encoding numeric values as ranges, and patterns as sequences of strings, using a protocol named SAX (Symbolic Aggregate approXimation). Both are then generalized to obtain k-anonymity. In this way, even though ranges become very wide thus losing too much information, characteristics of the time series are preserved in the pattern representation. I conjecture that the two representations could be concatenated and constitute a reasonable input for effective neural networks, but as I will argue in the next chapter, I believe there might be contexts in which considering the value of every observation as a quasi-identifier is too strong a requirement and leads to unnecessary data loss.

In [48], statistical matching techniques are taken into account and theoretical privacy achievability results are derived. Indistinguishability of time series, unlike many other definitions based on k-anonymity (with respect to the record values themselves), is here considered with respect to the empirical probability distribution estimated from the series. The attacker is assumed to know the real distribution and to use it for re-identification. This sounds like an idea with a lot of potential, but positive results are only shown for independent identically distributed (i.i.d.) data. It is not clear how this model should be applied to more realistic series, that anyway would have complicated probability distributions defined by very many parameters.

To conclude, while the techniques I have reported provide interesting stimuli, they appear not directly applicable for my purposes, either because their output is hard to manipulate, their definition of privacy is difficult to interpret or not relevant, the

perturbation of series falsifies the data inappropriately, or they have low utility since they do not use domain knowledge to relax anonymization.

Chapter 3

Theoretical framework

3.1 Problem definition

This work aims to obtain data anonymity through de-identification. In other words, it focuses on preventing the linkage between time series in a data release and the true identities of individuals that are described by those series. The objective of any good anonymization process should also be to remove the possibility of learning sensitive facts about individuals even when they cannot be fully re-identified. Let us consider the scenario of a data holder (or *custodian*) maintaining a database $d \in \mathcal{D}$, consisting of $|d|$ of multi-dimensional time series $x \in \mathcal{X}$. Each time series is a tuple of length $|x|$, where each tuple element $x^{(i)}$, for $i \in \{1, \ldots, |x|\}$, is called a *data point*. Time series of different lengths are allowed within the same database. A data point is itself a tuple with q elements $x_j^{(i)}$, called *records*, where q is constant for a database and represents the number of dimensions. Series can have *null records*, corresponding to a missing observation at specific time indices.

Optionally, the custodian can release a sequence of bijective functions $\tau_x : \{1, \ldots, |x|\} \rightarrow \mathbb{R}^+$. Each released time series would then have an associated function τ_x, which represents a mapping between tuple indices and *time indices*. In this way, defining T^x as the co-domain of τ_x (i.e. the times for which x has recordings), then we have

$$\forall i \in \{1, \ldots, |x|\} : T_i^x = \tau_x(i) \tag{3.1}$$

Even though time indices can be identifying, for the moment let us assume that is not the case, or that they are not provided to adversaries. This restriction will be lifted in practice in Chapter 5.

Let $\mathcal{P}$ represent a population such that all of its elements are individual identities. There exists a set $\mathcal{P}_d \subseteq \mathcal{P}$ such that there is a one-to-one mapping[1] between each element in $\mathcal{P}_d$ and entries of d. For each series x, the identity of its originator is denoted by P^x. The objective is to produce a *data release* $\tilde{d}$ from a time series database d such that the following conditions are satisfied (of which 2 and 3 are here only sketched):

[1] I make the simplifying assumption that there are no two series in d that belong to the same individual. Later sections concisely mention ways to lift such assumption.

1. $\tilde{d}$ has the same structure as d, and there is a one-to-one mapping between each $x \in d$ and their anonymized version $\tilde{x} \in \tilde{d}$.
2. For a constrained class of adversaries $\mathcal{A}$, the probability that an adversary $a \in \mathcal{A}$, modeled as a probabilistic algorithm, can output a reconstructed $\hat{x} \approx x$ after being given $\tilde{d}$ and any identity P^x is small.
3. With respect to a *utility loss* function c, the value of $c(\tilde{d}, d)$ is small.

These conditions will be made more concrete later. In particular, I build towards a description of how adversaries are constrained and what it means for the reconstruction probability to be small in Section 3.2 and 3.3. As for utility loss, in this report we look both at intrinsic measures (formulated as data loss) and extrinsic ones (the degradation in the performance of predictive tasks which depend on time series). A good utility function is task-dependent and therefore the choice of c should depend on the instantiation of a more general anonymization framework. The evaluation of utility chosen in this project for MIMIC-III is discussed in Chapter 5.

3.2 Feature k-anonymity

Unlike in the snapshot privacy setting, here we deal with time series and not with database tables, and hence we have the extra complexity of having multiple observations at different time indices for each dimension. In theory, we could apply k-anonymity to this model naively, by flattening out the time series database such that each record (observation for a dimension at a time index) represents a column. In this way it is possible to record all the data in a rectangular (2-dimensional) table. Then we could apply k-anonymity by considering all record columns as quasi-identifier. It is evident, however, how this transformation blows up the number of dimensions such that applying any k-anonymization algorithm is going to scrape all useful information. The previous part of the project showed the difficulty of k-anonymizing with respect to 10 quasi-identifiers. Even considering just one time series dimension or observation type (e.g. heartbeat rate in MIMIC-III), there are typically hundreds of observations. Hence, the effect of the curse of dimensionality hits as hard as it could in this setting.

However, it could be argued that it would be unreasonable in some circumstances to consider all time series records as quasi-identifiers. MIMIC-III records vital measurements of patients in the Intensive Care Unit. It is more intuitive to consider the values of vitals at several points in time as the sensitive attributes we want to protect rather than the identifying characteristics. The other information in the database (like demographics) should be easier to obtain for most attackers than the highly-granular time series. The individuals having access to the latter are likely to be insiders in the institution that recorded the data, like doctors and nurses, and as a consequence of their position and limited number, the threats they might pose could be contained by means other than anonymization, e.g. legal and social incentives and data access policies [10]. On the other hand, especially in light of the explosion of medical wearable devices [21], it is reasonable to assume that some properties and patterns that characterize health data could be obtained outside of the Intensive Care Unit context. For example, the average heartbeat rate of an individual throughout 24 hours might be well-known, and

if it is not significantly altered by an acute condition active during the ICU stay, could be used to match an individual to its full MIMIC-III history of measurements. For this reason, here we aim to protect regularities in the time series data, in the form of *features*, in order to prevent re-identification.

Define the universe of features $\mathcal{F}$ as the set of all possible features $f : \mathcal{X} \to \mathbb{R}$. Effectively, a feature is just an arbitrary function that takes a time series and returns a real number. The type of attackers that the anonymization framework proposed here tries to protect against will be defined with the help of the concept of sets of features. Assume that each attacker a has access to a knowledge base d^*, identical in shape to a time series database d. The knowledge base contains series x^* with identities P^{x^*}, which are the link that we want to protect between x^* and x in the database d under attack. The attacker uses its knowledge base to estimate the feature values in d, and attempts to re-identify series x by gathering equivalence classes $C_{\mathcal{F}',d}(x^*) = \{x' \in d | \forall f \in \mathcal{F}' f(x') = f(x^*)\}$ for $\mathcal{F}' \subseteq \mathcal{F}$. For the moment, we naively assume that an attacker only guesses x by choosing uniformly at random one of the series from the equivalence class it gathers, i.e.: $a(P^x, d) \sim \mathcal{U}(C_{\mathcal{F}',d}(x^*))$. Then, in order to make the guessing probability small, we want all records to be hidden in a crowd of k individuals, where k is ideally a large number, i.e., we require that:

$$\forall x \in d : |C_{\mathcal{F}',d}(x^*)| \geq k \tag{3.2}$$

such that the guessing probability is at most $\frac{1}{k}$ for all series in a database. Equation 3.2 is the definition of *feature k-anonymity*. Hidden in this definition there is an assumption on what the value of $x^* \in d^*$ is, given each series x. It is appropriate to choose the strongest assumption, i.e. we assume that:

$$\forall x^* \in d^* \; \forall x \in d \; \forall f \in \mathcal{F}^* : P^x = P^{x^*} \implies f(x) = f(x^*) \tag{3.3}$$

which is all that matters when gathering similarity classes. Additionally, because we cannot compute equivalence classes with respect to the whole universe $\mathcal{F}$, we choose a subset $\mathcal{F}^*$ which we use to indirectly model the accuracy of the background knowledge of the attacker. $\mathcal{F}^*$ should represent the most effective set of features the attacker could use. In order to justify our choice, we need to assume that for any other subset of $\mathcal{F}$, we cannot obtain a smaller equivalence class that still contains the target series x, i.e. we assume the following:

$$\forall \mathcal{F}' \subseteq \mathcal{F} \setminus \mathcal{F}^*, \forall x \in d : |C_{\mathcal{F}',d}(x^*)| \geq |C_{\mathcal{F}_*,d}(x^*)| \vee x \notin C_{\mathcal{F}',d}(x^*) \tag{3.4}$$

This could seem like an arbitrary assumption, but might also be useful in practice. If a data custodian had the certainty that some particular feature could be exploited for matching time series to identities, the framework described up to this point constitutes a helpful innovation that can be applied for risk mitigation.

Notice that the naive application of k-anonymity is an instantiation of feature k-anonymity. To obtain that, all we need is a feature set $\mathcal{F}^*$ such that each $f \in \mathcal{F}^*$ maps a series to its value at a specific index and dimension, and if the input series is not long enough, simply to a marker constant.

3.3 Feature-similarity disagreement

The framework presented up to this point has some fatal flaws:

1. A realistic attacker will not be limited to selecting a series at random from an equivalence class and can exploit the similarities among several series to approximate its knowledge
2. If it is helpful to the attacker, it does not have to constrain itself to equality checks to form equivalence classes, but can relax them to include series which are similar.

Problem (1) is closely related to homogeneity attacks for snapshot k-anonymity. l-diversity tackles this by enforcing that equivalence classes have got at least l "well-represented" values, on top of having size at least k. An alternative definition, t-closeness, requires, given the distribution of values for a sensitive field in each equivalence class and the distribution in the entire database, that their statistical distance is bounded from above by a constant t. t-closeness represents an improvement over l-diversity for one important reason: having l distinct sensitive values in an equivalence class does not exclude that they could be semantically very similar, thus effectively rekindling the problem of homogeneity attacks. In the present setting this difficulty is overcome, because time series consist of real numbers (not categorical values), which have implicit semantics. We pick an apt definition of diversity that captures the semantics of time series as their embedding in a high-dimensional space, and thus characterizes semantic similarity or difference as distance in such a space. Consider for example, given one-dimensional series y and z with the same number n of observations, Euclidean distance:

$$\delta_{Euclidean}(y,z) = \sqrt{\sum_{i=1}^{n}(y_i - z_i)^2} \tag{3.5}$$

A potential adaptation that is close to l-diversity in spirit, requires that the average Euclidean distance between pairs of series (*consistency*) in the same equivalence class is at least l for all classes. Diversity within equivalence classes adds useful uncertainty (from the point of view of a privacy-aware data custodian). This can however be further refined by realizing that in principle, we do not mind many series being similar among themselves, as long as they all are dissimilar from the target series, or they *disagree* with it. Let us then denote *disagreement* given a target series x and a series set S, as the following quantity:

$$\Delta(x,S) = \frac{1}{|S|}\sum_{i=1}^{|S|}\delta(x,S_i) \tag{3.6}$$

where all the series in S and x have the same length n and δ is a distance measure for multi-dimensional series.

I will not give a rigorous description of what it means for the probability of an attacker reconstructing x to be low in this case (unlike I have done in the previous Section when discussing feature k-anonymity). Even after restricting the attacker's background

knowledge to consist only of time series features, formalizing the probability of reconstructing x would require arbitrarily restraining the range of operations the attacker might perform. For example, the attacker could gather an equivalence class and believe x is close to the average of series in that class. Alternatively, it could sample a series from the equivalence class with probability proportional to the agreement with other series within the class. The attacker could also deploy any number of more sophisticated techniques. Nevertheless, I informally argue that a high disagreement will be conducive to better privacy protection against re-identification. For example, with the first example technique the probability of re-identification decreases because more disagreement leads to the average series being distant from the true x. With the second example technique, large enough disagreement means that there are several series which could be sampled that differ significantly from the true x.

Turning to Problem (2), it is not immediately evident why an attacker would be interested in enlarging an equivalence class. To see that, note that the feature values of the series x to reconstruct might have been incorrectly estimated with a small error, and therefore x resides in a neighboring equivalence class. In fact, it would be surprising if an attacker were able to match with strict equality features in its knowledge base and those of the slightly different context of ICU. To rephrase the above: while it is true that making an equivalence class could introduce uncertainty, it can introduce useful uncertainty (from the point of view of the attacker), for example by including the target series among those being considered. What is more, the overall uncertainty could actually decrease. For example, by swapping equality with similarity, a large number of very similar series could be taken into consideration, which decrease the previous uncertainty produced by dissimilar time series values in the same equivalence class. That is, *consistency* is increased. Consider now attackers gathering *similarity classes*:

$$C_{\mathcal{F}^*,d,t}(x^*) = \{x' \in d | \forall f \in \mathcal{F}^* : f(x') \approx_t^* f(x^*)\} \tag{3.7}$$

where $\approx_t^*$ is the similarity operator used by the attacker, parameterized by a distance threshold t. Again, we assume that $f(x^*) = f(x)$. For the remainder of this report, this definition of $C_{\mathcal{F}^*,d}(x^*)$ supersedes the one presented in Section 3.2.

Turning to the parameter t, the question arises of which value the attacker might use. The answer is any, depending both on random or hard-to-model characteristics of the attacker, and on opportunity. The value t might thus be adjusted on a case-by-case basis, depending on what group of series will "convince" the attacker that it is getting closest to the truth. Imagine, given a series x, plotting disagreement as described previously, as a function of the threshold t. This will from now on be referred to as *disagreement plot*, or *curve*, illustrated in Figure 3.1). We could quantify the privacy risk as the global minimum of this curve. However, since the attacker does not know x, it cannot compute disagreement and will have to take a guess. For this reason, a more meaningful measure could be the area under the curve (call it Δ-*AUC*) with respect to the plot just described:

$$\Delta\text{-AUC}(x, d, \mathcal{F}^*) = \int_0^{t_{max}} \Delta(x, C_{\mathcal{F}^*,d,t}(x^*))dt \tag{3.8}$$

However, the area under the curve is in some ways a poor summary of the full plot, as it

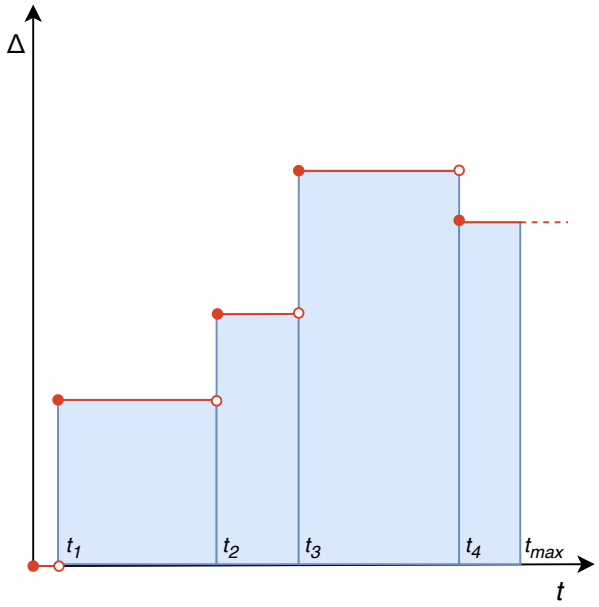

Figure 3.1: Hypothetical plot of a disagreement curve as a step function (in red) with the area under the curve colored in light blue.

can be misleading in the case where disagreement is extremely high for large distance thresholds but very low for small ones. Finding a unique summary of disagreement that could functionally replace in this setting point-estimates of privacy (like k-anonymity) is outside of the scope of this project. On the other hand, it can be argued that the disagreement framework introduces useful language for the discussion of the novel privacy considerations made, and the disagreement plot is a useful tool to visualize the level of privacy protection offered by a database for several similarity increments and for different series, and as will be seen later the plot can help guide the anonymization process.

It is perhaps interesting to notice that, while feature k-anonymity is a generalization of k-anonymity, feature-similarity disagreement is not. In fact, even though equivalence classes could be considered a special case of similarity classes, the distances that disagreement is built on are computed with respect to secret time series, rather than on the public (de-identified) series considered by k-anonymity. This argument is in relation to disagreement evaluated at a single similarity threshold. Summaries of the full curve (like Δ-AUC) add further incompatibility which does not allow an instantiation of the framework to equate k-anonymity or any of its known extensions. Another difference is that, unlike k-anonymity, disagreement is computed for individual series, even though the disagreement for many series can be aggregated. Finally, note that even though disagreement is based on distances among series and k-anonymity is based on counts, if considering only equivalence classes it is possible to come up with a series distance measure that relates k and disagreement. For example, if series distance is defined as being 0 when series have identical feature values and 1 otherwise, then feature k-anonymity holds if and only if disagreement is not greater than $\frac{|d|-k}{|d|}$ for all $x \in d$.

Earlier we assumed there is no more than one series per identity in d. If this was not the case, a new metric could be devised by combining the re-identification risk for each series coming from the same individual. This can also be extended to cover the case in which you have individuals so similar that their series are expected to lie in the same spot of feature space, their underlying series are themselves very similar, and such similarity is public knowledge (as could be the case for twins). In this case it suffices to combine the re-identification risks for each related individual.

Does knowing whether a particular individual P^x appears in d benefit an adversary? If the attacker is not sure, it can either try to match a series that appears in the database, or one that does not. In the first case, there is even more uncertainty than that modeled by disagreement. In the latter case, whatever the attacker infers is not modeled as a direct risk for the series in d. Assume now an attacker knows a series x it is trying to match is in d. The risk of a re-identification attack is precisely what disagreement seeks to model. Without attempting to match any series, the attacker might know an individual is a member of $\mathcal{P}_d$ and learn what it can on the basis of the characteristics common to all series in the database. This is taken into account with the summation in the definition of Δ-AUC and specifically for t_{max}, the largest possible value of t (which is a parameter to set).

Consistently with the problem definition in Section 3.1, features must be able to accept input series containing null records. If all the records for a dimension are null, the feature value is itself null. Null feature values do not leak privacy, i.e. the absence of information is assumed to be non-identifying. Additionally, because null feature values are assumed not to carry meaning, equivalence classes cannot be defined by null feature values, because equality to unknown values is undefined. We can go round this issue in the context of feature disagreement by utilizing a feature distance measure $\approx_t^*$ that, when receiving two series y and z, compares only the feature values that are concrete in both series. Similarly, we need to use a series distance δ that can deal with some null records.

Chapter 4

Algorithmic implementation

4.1 k-anonymizing features

The disagreement metric is effective in disentangling the measure of privacy from assumptions over similarity classes that could be used for re-constructing series properties, but as will be seen in Section 4.3, it is expensive to compute. For this reason, it is more convenient to maximize it indirectly and verify the privacy gain after anonymization, rather than using it as an objective function of an iterative anonymization process. This is what has been done and explained in Chapter 6, with experiments both aimed at gauging the effectiveness of the proposed algorithms, and to illustrate the practical procedures that a custodian might apply with respect to said algorithms.

It must be noted that low values of disagreement for high distance thresholds mean that not much can be done to protect the database from attackers that know who took part in it, except adding falsifying noise to records. On the other hand, if the database possesses a pool of disagreement, then it is possible to distribute it so as to raise the score for lower thresholds. For example, it is possible to move records closer together in clusters of similar features such that for low distance thresholds, those clusters will hopefully be diverse in terms of series values. This concept is illustrated in Figure 4.1. Of course, if the combination of features is too rich, or highly correlated with series values, disagreement gains are again hard to achieve.

Anonymization under this framework is thus not a trivial matter. However, creating clusters of series with similar features seems a desirable starting point. One way to do so is k-anonymization. Many k-anonymizing algorithms have been proposed [45, 23, 39]. In the previous part of the project, one called Optimal Lattice Anonymization (OLA) [9] was chosen because it was the most computationally efficient among those that find a globally optimal solution with single-dimensional global recording, with respect to a monotonic utility function of choice. What single-dimensional global recording entails is explained below. Here, however, our needs are slightly different. First and foremost, we are not dealing with categorical values, but with real numbers. OLA needs a user-defined generalization hierarchy. While it is possible to craft a generalization hierarchy for numeric values, by converting values to ranges of increasingly larger

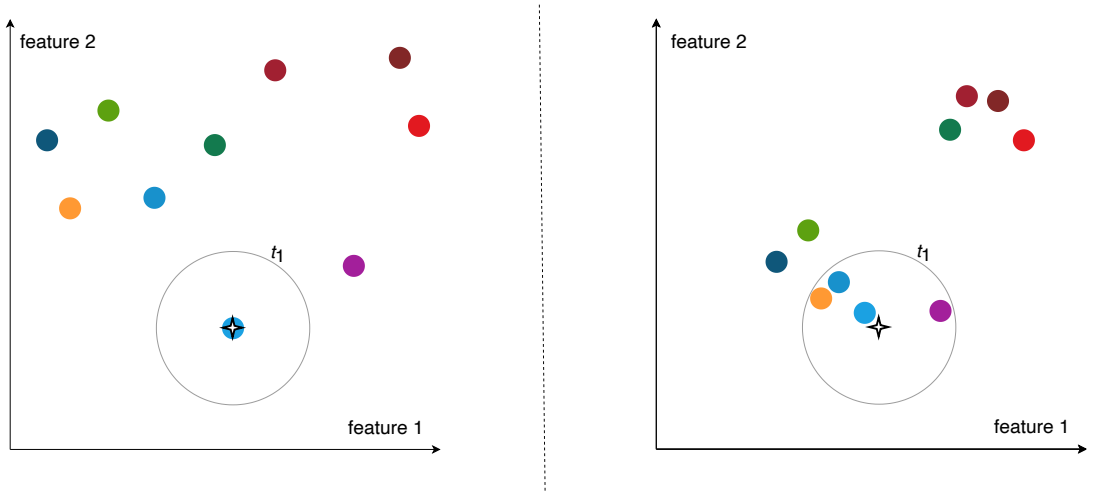

Figure 4.1: On the left, series plotted in a two-dimensional feature space, with similar colors representing similar series values. On the right, how clustering together series based on feature values increases disagreement with respect to a series x (marked by a four-points star), at a distance threshold t_1. Notice that disagreement is calculated on the basis of the original series, and therefore in the image on the right the star and blue circle do not overlap. If we managed to achieve feature k-anonymity, series belonging to the same feature equivalence class would all be stacked in the same point in feature space.

width, this is a slightly unnatural approach. Having too many generalization steps makes OLA prohibitively slow, and having too few means we are missing some optimization opportunities with respect to utility preservation.

On the other hand, the algorithm Mondrian [24] is perfect for dealing with real values, and even though it does not find an optimal solution, it allows to look in the space of multi-dimensional local recordings, thus in principle (and often in practice) allowing to find solutions with higher utility. Mondrian is a top-down greedy algorithm that starts from a solution in which all data entries are located in the same giant equivalence class, which is then recursively subdivided until any further equivalence class split would break the k-anonymity guarantee. The operation of Mondrian is described more precisely in Algorithm 1.

The dimension on which to split and the particular split are both chosen through heuristics. As it is common, in my implementation values are split on the median. As for the choice of dimension, they could be split on the dimension with largest range of values (*range heuristic*), or dimensions could be ordered on the basis of the maximum achievable average pairwise distance between all entries in the resulting split (*diversity heuristic*). In this way we would be penalizing splits along dimensions that risk resulting in too much agreement. After identifying desired equivalence classes, their quasi-identifiers are generalized. Instead of generalizing numbers to ranges, I chose to output the class mean, which can more easily be the input of machine learning models. Notice how, by convention, null feature values end up in the right partition. They will subsequently be ignored (and preserved) when generalizing members of

Algorithm 1 Mondrian multi-dimensional k-anonymity

```
function anonymize(partition)
    if not can_split(partition) then
        return {partition}
    else
        dim ← choose_dimension(partition)
        splitVal ← choose_split(partition,dim)
        lhs ← {t ∈ partition : t.dim ≤ splitVal}
        rhs ← {t ∈ partition : t.dim > splitVal ∨ t.dim = null}
        return anonymize(rhs) ∪ anonymize(lhs)
    end if
end function

equivalenceClasses ← anonymize(database)
release ← generalize(equivalenceClasses)
```

partitions. They will then be neglected by the constraint optimization algorithm from Section 4.2.

It is worth briefly explaining how different types of recordings affect k-anonymization. Given a generalization function (corresponding to a hierarchy), global recording [50] entails that such a function can be applied to either all or no records in the database. The same restrictions do not apply to local recording. We can further subdivide recordings in single-dimensional and multi-dimensional [23]. In the first case, generalization functions have *arity* equal to 1, i.e. they only take one argument (corresponding to a single table field) and output one value, representing the generalized field value. Naturally, as we are dealing with functions, all fields having the same values to which the same generalization rule is applied will be mapped to the same generalized values. On the other hand, in the case of multi-dimensional recordings, generalization functions have an arbitrary arity, and map the cartesian product of several field values to their generalization. This means that it is allowed for single fields with the same value across rows to have different generalizations, as long as the cartesian product of all the values involved in the generalization is consistently mapped to the same output. While global single-dimensional recordings maintain a higher level of recording consistency and simplify several data analysis tasks [9], I have chosen an approach that could, in some cases, preserve more information, benefitting the predictive capability of models trained on the anonymized data.

As a final note, it is perhaps interesting to notice that traditional k-anonymization strategies cannot be reused effectively to obtain approximate feature k-anonymity. As features are arbitrary functions, we are not guaranteed that generalizing the underlying series will have any impact (or a desirable one) on the feature values. See Figure 4.2 for a representation of this problem.

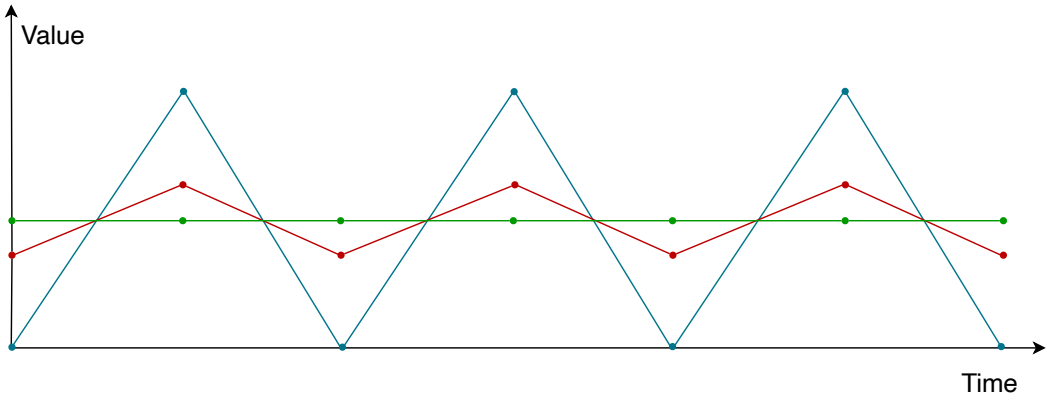

Figure 4.2: A time series with 7 records (in blue) and possible generalizations of its values (in red and green, in order of how extreme the generalization is). Notice that with respect to the feature "average distance between peaks", the blue and red series are identical, whereas all information is lost in the green one, i.e.: there are no peaks, and feature k-anonymity is obtained at a disastrous price for utility.

4.2 Constraint optimization

Up to this point, an important aspect has been ignored: it is impossible to change the value of features without altering the underlying series. This means that once we identify the desired (or *target*) features, we need to identify the corresponding series perturbations that allow us to achieve the targets. It is within the realms of possibility that there is no way to obtain all the target features at the same time; in fact they might theoretically impose conflicting or even opposite transformations. We hope however that we can get close to them. We also hope that transforming the series to meet the targets does not destroy too much utility. Since new feature values come from the output of the Mondrian algorithm, which clusters similar features together, the required perturbation should not be too extreme. It must be highlighted that perturbing the underlying series will to some extent increase disagreement on its own, because disagreement is measured with respect to the difference with the original series. The perturbation that we will look at is additive: we search for a sequence $\bar{x}$ such that, given a set of features $\mathcal{F}^* = \{f_i\}$ and target values $\{v_i\}$, we have that

$$\forall i \in \{1,...,|\mathcal{F}^*|\} : f_i(x + \bar{x}) - v_i \approx 0 \tag{4.1}$$

We then release (or measure the disagreement) of $\tilde{x} = x + \bar{x}$.

Finding the appropriate $\bar{x}$ is an optimization problem. In the interest of making the framework as general as possible, no limitations are imposed on the choice of feature functions, for example requirements of differentiability or continuity. This means that most traditional optimization strategies are precluded. For instance, hill climbing algorithms [37] require double differentiability, whereas Newton's Method [11] needs convex functions. For this reason I have resorted to a genetic algorithm [31]. Genetic algorithms are inspired by processes of biological evolution and natural selection and are applicable to generic search and optimization problems, allowing for a great flexibility, while generating solutions of much greater quality than more primitive

gradient-less methods like random search. The operation of a genetic algorithm starts with the random generation of an initial set of solutions. Then, iteratively, it applies the following three *genetic operators*:

- **Selection**: A subset from the previous set of *chromosomes* (candidate solutions) survives the iteration, in such a way to privilege ones that have greater *fitness* with respect to a fitness function.
- **Crossover**: Offspring is generated by combining *genes* (characteristics) of selected chromosomes.
- **Mutation**: The new offspring is subject to some stochastic transformation, thus allowing to move in the search space.

The quality of a chromosome is judged on the basis of an arbitrary, user-defined user function. We obviously care about keeping the perturbation low in magnitude. Fitness should thus penalize solutions where the sum of the absolute value of genes is large. Additionally, solutions should be such that we obtain the target feature values. This can therefore be formulated as a constrained optimization problem. Rejecting all solutions that do not precisely meet the constraints imposed by the target feature values would however be very wasteful. For this reason, I attempted to employ a technique called *annealing* as proposed by [3], that balances the distortion minimization objective and the feasibility condition in a dynamic manner, as a function of the iteration number. In initial iterations the first objective is given a larger weight so that the algorithm can explore the space of unfeasible solutions while looking for a global maximum. As time progresses, the fitness score for infeasible solutions will tend to zero. Consider vectors of length n, $\boldsymbol{f}$ and $\boldsymbol{v}$, comprising feature and target values, respectively. The exact computation of fitness at iteration m is:

$$\text{objective}(\bar{x}) = -\sum_{i=1}^{|\bar{x}|} |\bar{x}_i| \tag{4.2}$$

$$\text{feasibility}(\bar{x}, x, \boldsymbol{f}, \boldsymbol{v}) = -\sum_{i=1}^{n} \sum_{j=1}^{|\bar{x}|} |f_i(x_j + \bar{x}_j) - v_i| \tag{4.3}$$

$$\text{fitness}(\bar{x}, m) = \text{objective} \cdot \mathrm{e}^{-\text{feasibility} \cdot \sqrt{m}} \tag{4.4}$$

In my implementation, an initial population is generated from a normal distribution with zero mean and tunable variance. Then, at each iteration n candidate solutions are retained using a modified version of the popular *tournament selection* heuristic [30]. For as many new chromosomes as we desire, we select two parents by sampling g (the size of the tournament) previous chromosomes uniformly at random without replacement, and picking the two with the best fitness. In classical tournament selection, if g is smaller than the number of chromosomes, it is possible that the fittest chromosome will not pass on its genes, but this also prevents prematurely getting stuck in a local optimum. In my adaptation I additionally always keep the best chromosome, without having it participate in crossover or mutation. Parent candidate solutions are crossed over by selecting uniformly at random crossover point such that at least one gene from

each solution is included, and then concatenating the left section of the first parent and the right section of the second parent as split by the crossover point. Before adding the new solution to the candidate set, it is mutated by sampling multinomial Gaussian noise with zero mean (the variance is a parameter that can be tuned). The multinomial noise can, for each gene, either be added at the corresponding position, or be ignored, on the basis of binary random variables with a Bernoulli distribution, whose probability is another parameter to tune. After at most a fixed number of iterations, the best candidate solution is returned. For computational efficiency, if no improvement on the fitness function is detected for longer than a set number of iterations, the algorithm returns early.

Although evolutionary algorithms are also used by InPaCT [22] (see Section 2.4), it is useful to notice that there is a crucial difference in my approach: InPaCT looks for perturbations such that the released series, that attackers are able to identify, have characteristics that satisfy some constraints, that are functions of the input and output series. The constraints are used to protect some secrets of the true series. In here, we look at perturbations that make characteristics for specific individuals hard to derive because the attacker receives series with inconsistent values and has a hard time distinguishing between them. The formulation of the problem as constraints-satisfaction is only instrumental to de-identification.

4.3 Computing disagreement

Let us now look at how to compute disagreement and Δ-AUC. The integral in t (distance threshold) can be transformed into a summation by noticing that, for each $x \in d$, there are at most $|d| - 1$ other distinct combinations of feature values. Hence, we can just look at threshold increments such that for each one we include at least one more series in the resulting similarity class. Indeed, if Mondrian and the genetic algorithm were able to create n feature equivalence classes, there would be exactly n thresholds, each one merging together nearby equivalence classes into larger similarity classes. Unfortunately, the output of the genetic algorithm is expected to be imperfect, and thus we need to consider up to as many thresholds as there are series. In any case, notice that the disagreement function is a step function. The value of disagreement remains constant until we hit a new threshold that allows a new series to join the similarity class. Therefore, to calculate the area under the curve, we simply sum the areas of the rectangles of width $t_{i+1} - t_i$ for each i and height equal to the disagreement value.

The complete procedure is described in Algorithm 2. The first two lines construct $|d| \times |d|$ (non-symmetric) matrices of features and series distances between original series x and released series $\tilde{x}$. If instead of a de-identified release the series being assessed are the original ones (with features matching the adversary's expectation), the distance matrices will be symmetric. This step is quadratic in the number of series. The outer loop iterates over series to compute disagreement for each of them, and also to look for the minimum Δ-AUC value. The function create_increments sorts all the distances from x and groups other series by those distances. Hence, the outer loop in

Algorithm 2 Computation of disagreement statistics

Fill *featureDist* matrix
Fill *seriesDist* matrix
Initialize empty *allCurves* object
$min \leftarrow \infty$
for x in d **do**
    allCurves.add_curve()
    $auc \leftarrow 0$
    $summation \leftarrow 0$
    $normalization \leftarrow 0$
    $thresholdIncrements \leftarrow$ create_increments($seriesDist[x]$)
    for t_i,$neighbors_i$ in *thresholdIncrements* **do**
        $summation \leftarrow summation +$ sum($seriesDist[x][neighbors_i]$)
        $normalization \leftarrow normalization + |neighbours_i|$
        $value \leftarrow summation/normalization$
        $auc \leftarrow auc + (value * (t_{i+1} - t_i))$
        allCurves.add_point($value$, t_i)
    end for
    allCurves.end_curve(t_{max})
    if $auc < min$ **then**
        $min \leftarrow auc$
    end if
end for
return *min*, *allCurves*

the algorithm takes $O(|d|^2 \log |d|)$ [1]. This dominates the running time of the algorithm which clearly does not scale well with large databases. Hence, this should not be used as a cost function to iteratively optimize, but as an *a posteriori* performance evaluation. The disagreement at each threshold is not recomputed from scratch at each iteration, but it is accumulated iteratively.

Because it is likely that the de-identified feature values of x are different from its original ones, it is possible that disagreement is undefined between $t_0 = 0$ and t_1, because no feature values are similar enough to the expected ones for x. In this case we simply disregard that portion of the disagreement curve. Finally, as it has been mentioned, t_{max} needs to be picked. If the largest feature distance between x and any other series were to be chosen as t_{max}, we would disregard the disagreement of the class comprising the entire database. Therefore, by convention I pick the largest feature distance, plus the average feature distance between x and all other series.

I used a measure inspired by Euclidean distance for both feature and series distance because it is efficient to compute. Any other metric could be used though, and in particular some might be more appropriate for the distance between series. For example, Minimum Jump Cost (MJC) and Dynamic Time Warping (DTW) are both shown to be more effective than Euclidean distance in several classification tasks [40]. They are both computed using a dynamic algorithm, by finding a likely alignment between different series records. It is possible to swap them in to measure disagreement if we want to focus on the possibility that an attacker might perform classification on the series we want to protect. Euclidean distance was adapted so that it ignores the positions in which either of its two inputs has a null value. Additionally, for series distance, in order to be able to deal with multi-dimensional series, it sums over each series index and each dimension.

[1] assuming that sorting takes $O(|d| \log |d|)$

Chapter 5

Application to MIMIC-III

5.1 Identifying features

With no claim to present an entirely comprehensive and accurate estimate of the most likely sources of privacy hazards, here I present a short analysis of health features that could be obtained and used for re-identification, collected by mobile consumer products connected to smartphones and the internet. I argue that an attack from a malicious agent providing health tracking, or from a hacker compromising the data recorded in mobile devices, is in today's environment realistic. Even popular fitness trackers such as Fitbit have been shown to have critical vulnerabilities, such as weaknesses in the security of the encryption and transmission channels [53]. This analysis serves as the basis for the example instantiation of the anonymization procedure and measures that will be evaluated in Chapter 6. The sample average, sample variance, minimum and maximum values have been chosen as features along the following high-risk dimensions:

- **Heart rate**: almost all smart watches in the market can be equipped with heartbeat monitoring capabilities [36]. For this reason, I expect this to be one of the most sensitive dimensions. Phenomena such as heart rate variability (measured by the variation in the beat-to-beat interval) could be highly identifying. However, MIMIC III does not record data that is as precise as what might come from an electrocardiogram. Instead, it records the heart rate at different points in time.
- **Systolic and diastolic blood pressure**: blood pressure recording is another staple of fitness trackers.
- **Oxygen saturation**: the ratio between oxygen-saturated hemoglobin and total hemoglobin is considered an important indicator of fitness and health and pulse oximetry is a non-invasive proxy to its measurement. Most smart watches allow to record this quantity.
- **Glucose level**: monitoring the level of glucose in blood is crucial for the management of diabetes. Increasingly sophisticated devices have been developed for this task [4]. The most common way of measuring it requires inserting a small quantity of blood into a test reagent strip. This means that this measurement cannot be performed by general-purpose smart watches. However, there

are commercially available meters that synchronize measurements with health tracking mobile apps.

Height and weight of patients are included as time series with very few measurements (often only one). Their sample mean was modeled as a sensitive feature. The dimension "Mean blood pressure" has been found to be highly correlated with the sum of "Diastolic blood pressure" and "Systolic blood pressure" (even though not equal to the actual mean). Therefore, this dimension has been dropped altogether. Within this privacy model, that again focuses on background knowledge obtained by consumer health trackers, I have deemed all other dimensions to constitute lower risk and thus features have not been extracted from them. The following two dimensions deserve a special mention.

- **Respiratory rate**: it has been shown [15] that it is possible to infer it when the body is resting, with *some* degree of accuracy (by measuring the physical transmission to the limbs and wrists with an accelerometer). Nevertheless, wearable devices are not in any position to measure directly and reliably the respiratory rate (measured as the frequency by which the chest rises).
- **Temperature**: it appears that, to date, some of the most popular fitness trackers do not record body temperature. One advertised reason is that the skin temperature at the wrist only allows for inaccurate measurements. Even though it would not be hard to conceive in the near future smart body thermometers that communicate with mobile phones, they would probably be used to measure unusual temperatures, rather than for constant monitoring.

It is easy to come up with many alternative privacy models, and their choice should be informed by expert knowledge. For example, one might assume that whether a patient entered a coma, or had a fever, is a known fact. In this case one can define binary features on the dimensions "Glasgow Coma Scale" and "Temperature", respectively. Similarly, more descriptive feature types than the four used in the following experiments can be utilized. Unfortunately, with the present definition of feature, there is no natural way to model the uniqueness of short sequences of highly identifying values. To do so would require having a binary feature indicating the presence of a sequence for all possible sequences. Even disregarding the very large number of features, the optimization problem described in Chapter 4 is not an effective way to suppress short unique sequences. Finally, note that with the sensitive characteristics of time series in mind, listed in [55] (see Section 2.4), no feature describing the trend and periodicity of series has been extracted and considered in this analysis.

5.2 Full-pipeline overview

The data transformation and evaluation pipeline that was designed is schematized in Figure 5.1. The original data source are tables from MIMIC-III. The code for the first steps of the pipeline was publicly released by [16] and has been re-used in this project. The benchmark code generates time series from the MIMIC-III tables, and divides them in different sets for model training and evaluation. For the purpose of benchmarking non-anonymous data, the data would be fed to a specified model among the baselines

provided by the benchmark, and performance statistics would be reported. Instead, the pipeline has been extended with de-identification steps. Furthermore, before evaluating any model or anonymizing, I added a further pre-processing step, to aid both training and de-identification. Pre-processed series have then been saved on disk and iteratively loaded, de-identified, and finally evaluated.

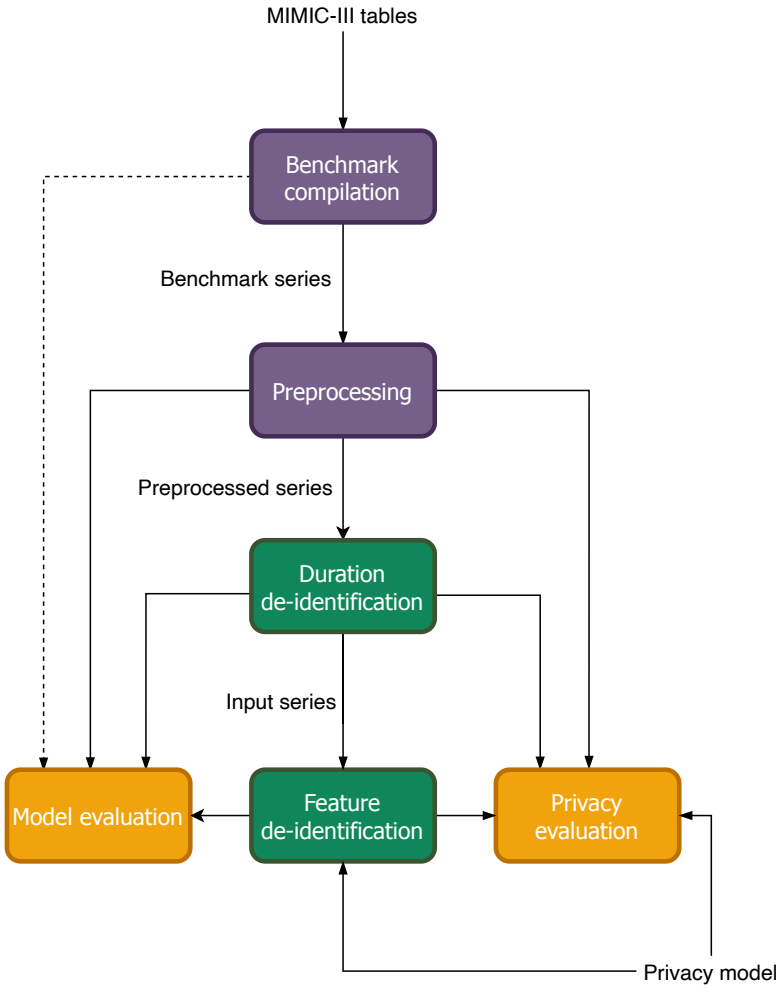

Figure 5.1: Overview of the data processing pipeline. Some details are omitted and are be illustrated by the Figures that follow. In purple, the data generation procedures, outputting potentially identifiable data. They are run only once and their output is saved on disk. In green, the data de-identification procedures, run and evaluated iteratively (through the procedures represented by orange boxes). The dotted line represents the model evaluation performed in [16], and not replicated in this project.

5.3 Time series generation and pre-processing

The benchmark compiler loads the tables corresponding to patient stays, *lab events* and *chart events*. It first groups relevant information by patient, discarding individuals younger than 18 years so that the model does not have to deal with the rather different

characteristics of pediatric physiology. Patients with multiple ICU stays or transfers across wards are also dropped. Time series are then compiled, each one of them being a $samples \times dimensions$ table, where the 17 dimensions correspond to physiologic variables. Each variable comes from different types of events combined, and is transformed so that they are consistent. e.g. in terms of units of measurement.

The pre-processing routines begins by dropping the columns specified in the previous chapter. Even with the variable transformations performed by the benchmark code it turns out that, for some dimensions, values still appear as if they were recorded using different systems or units. To slightly ameliorate the problem I convert to Celsius all temperatures that were almost surely recorded as Fahrenheit (i.e. those whose scalar value is larger than 85). All records have been encoded as floating-point values. This is in contrast with the benchmark data readers, that consider four dimensions, corresponding to attributes on the Glasgow Coma Scale, as categorical values. Because these values have a total ordering, corresponding to increasing patient responsiveness, classifiers could use numerical encodings advantageously. Additionally, this simplifies the resampling and interpolation procedures that I am about to describe. Receiving input with recordings at consistently-spaced time intervals generally helps sequence classifiers, as mentioned in Section 2.1. It also allows to drop the time indices entirely, which supports the de-identification of durations, dealt with in the following section. All series have then be resampled hourly, starting from the time of their first recording. New values have been generated using linear interpolation.

5.4 Duration de-identification

Given the one-hour resampling interval, it is not possible to identify series based on their sampling rate. However, it might be possible to re-identify individuals using the number of observations (series *duration*), used as a proxy for the length of stay in the Intensive Care Unit, in hours. For the remainder of this report, consider adversaries knowing the length of stay of patients. This strong assumption is crucial to protect against a class of attacks analyzed in Section 6.1. Hence, I assigned series to different duration classes. The goal is to have duration classes as large as possible, so as to increase their diversity and improve disagreement. I implemented a procedure, described in Algorithm 3 to merge together different duration classes so as to enforce that all of them have size at least k. The algorithm works by assigning series to their original duration classes and shifting them to shorter durations if that helps obtaining classes of size k, either because of the deletion or the enlargement of small classes. Series will always be moved to the closest shorter duration class that needs more entries. Once enough series are re-arranged to make all classes have size k, leftovers are re-assigned to their original classes. Thanks to these two latest features, the number of re-arranged series is minimized. After series are assigned to a class, they might need to be shortened to reflect the assignment. While this can be done in many ways, I chose to trim them from the beginning, because for the in-hospital mortality prediction task, used for utility evaluation, the final part of time series is the most meaningful.

In principle, it might be possible to reconstruct times of day by exploiting knowledge on the physiologic periodicities of some vital signs. However, this scenario has not

been considered here. Series are assumed to not contain any revealing time information except their length, and thus they conform to the theoretical model (and representation) outlined in Chapter 3.

Algorithm 3 De-identification of lengths

```
Initialize byLength mapping
Assign series indices to length in byLength
Initialize lengthByIdx mapping
Assign lengths to series indices in lengthByIdx
Initialize newByLength mapping
Initialize queue
for l in reverse_sort(keys(byLength)) do
    for i in byLength[l] do
        append(i, queue)
    end for
    if length(queue) < k then
        continue
    end if
    while length(queue) > 0 and length(newByLength[l]) < k do
        append(pop(queue), newByLen[l])
    end while
end for
while length(queue) > 0 do
    i ← pop(queue)
    l ← lengthByIdx[i]
    append(i, newByLen[l])
end while
newByLen ← drop_rare_lengths(newByLen, k)
newSeries ← shrink_series(newByLen)
return newSeries
```

5.5 Feature de-identification

After creating duration classes, the feature de-identification algorithm has been applied to each duration class separately, and thus their privacy will also be analyzed disjointly. A schematic representation of the feature de-identification process is presented in Figure 5.2.

According to the privacy model described in Section 5.1, the only types of features are the mean, variance, minimum and maximum values of series. With this in mind, it is possible to design an ad-hoc constraint optimization algorithm that looks for some appropriate additive transformation in a more direct manner than the genetic algorithm. The chosen algorithm proceeds as follows, iteratively:

1. Cut off all records below the target minimum and above the target maximum

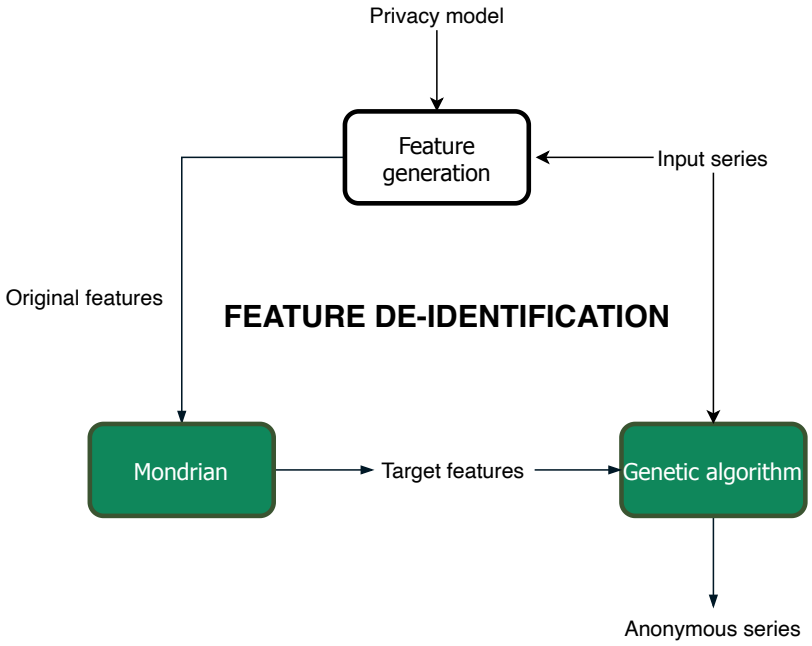

Figure 5.2: Overview of feature de-identification.

2. If necessary, increase the absolute values of the minimum and maximum records to meet the minimum and maximum targets

3. Subtract from each record the difference between the mean of the series and the target mean (shift the entire series up or down)

4. Add some small Gaussian noise to every record (to help with zero-variance series). Then divide each record by the ratio of the standard deviation of the series and the target standard deviation (stretch the series away from or towards its mean)

It is necessary to iterate because the last two steps modify the minima and maxima. However, in successive iterations, the shifting and stretching have lower magnitude and thus a smaller impact on any individual record. This algorithm works well in practice, requiring a lot fewer iterations than the genetic algorithm, but only works under the present privacy assumptions. It has been used, as described in Chapter 6, as a "gold standard" against which to compare the performance of the genetic algorithm, and also to speed up the optimization of Mondrian's parameters. It is interesting to compare the genetic algorithm against the ad-hoc algorithm for two reasons. Firstly, some constraints might be impossible to achieve by any algorithm, and it is useful to find a more realistic baseline result than the full satisfaction of all constraints. Secondly, I did not at first exclude that the great flexibility of the genetic algorithm could lead to better solutions.

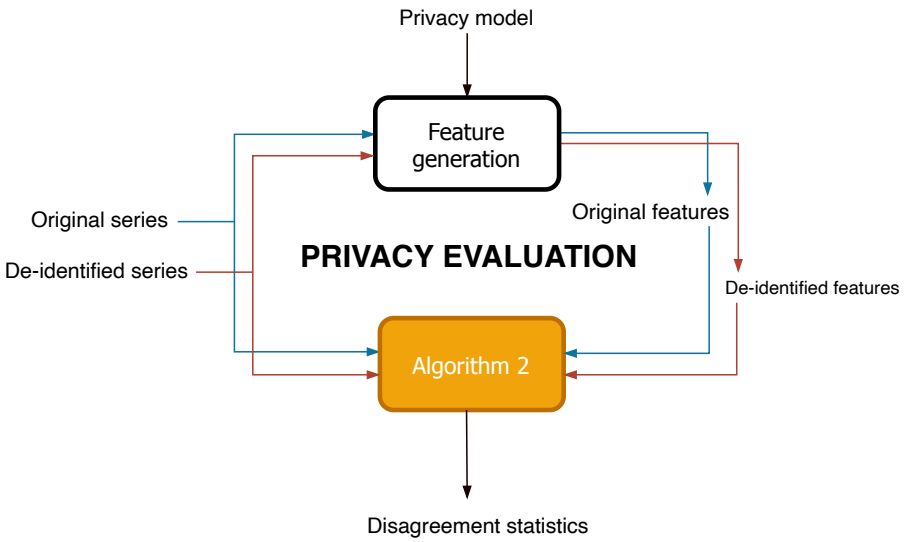

Figure 5.3: Overview of privacy evaluation.

5.6 Privacy evaluation

Disagreement values were computed as illustrated in Figure 5.3. As already explained, the area under the curve is only an imperfect synthesis of indistinguishability at different similarity levels. For this reason, full disagreement curves were inspected.

It is useful to use a visualization that summarizes all the curves for series within the same duration class. To do so, I plotted summary statistics (like minimum, maximum and mean values) at each threshold value. With the disagreement plot defined in Section 3.3, such a summary plot is rather hard to interpret, because of the large range of values the distance threshold can take on. For N series, the maximum distinct similarities between them is N^2, and most series will only have points for a subset of similarities from such range. In practice, this means that the curves on the summary plot will dramatically jump up and down. One way to address this is by smoothing the summary curves. Instead, the way I have chosen to address this was by looking at the percentage of series that entered a similarity class (*inclusion percentage*) when decreasing the distance threshold, rather than at feature distance directly. In this way, the x axis only has up to N points, and most series will have disagreement values for all the points. This change of unit also re-aligns series on the x axis. For some of them, a small distance threshold variation is sufficient for a large increment in the percentage of series in the class, while more feature-unique series will obtain the same increment in percentage at a much larger distance threshold. In a way, this makes the x axis more interpretable, by assigning it an intuitive explanation and relativizing it to the full dataset (or duration class), rather than tying it to a more obscure similarity value.

Two additional tricks can help reading disagreement figures. The first is using a logarithmic scale on the x axis. This reflects the importance of looking in a more granular manner at threshold increments when the feature similarity is high. The second trick is normalizing the disagreement values, or dividing all the curve points by the maximum disagreement at the latest distance threshold. In this way each value represents a ratio

between the actual disagreement and the best-case disagreement obtained when looking at the full dataset (or duration class). Note that after normalization, some disagreements might still be larger than one. An example of the resulting *summary disagreement plot* is illustrated in Figure 5.4, together with 100 of the original curves that generated the summary.

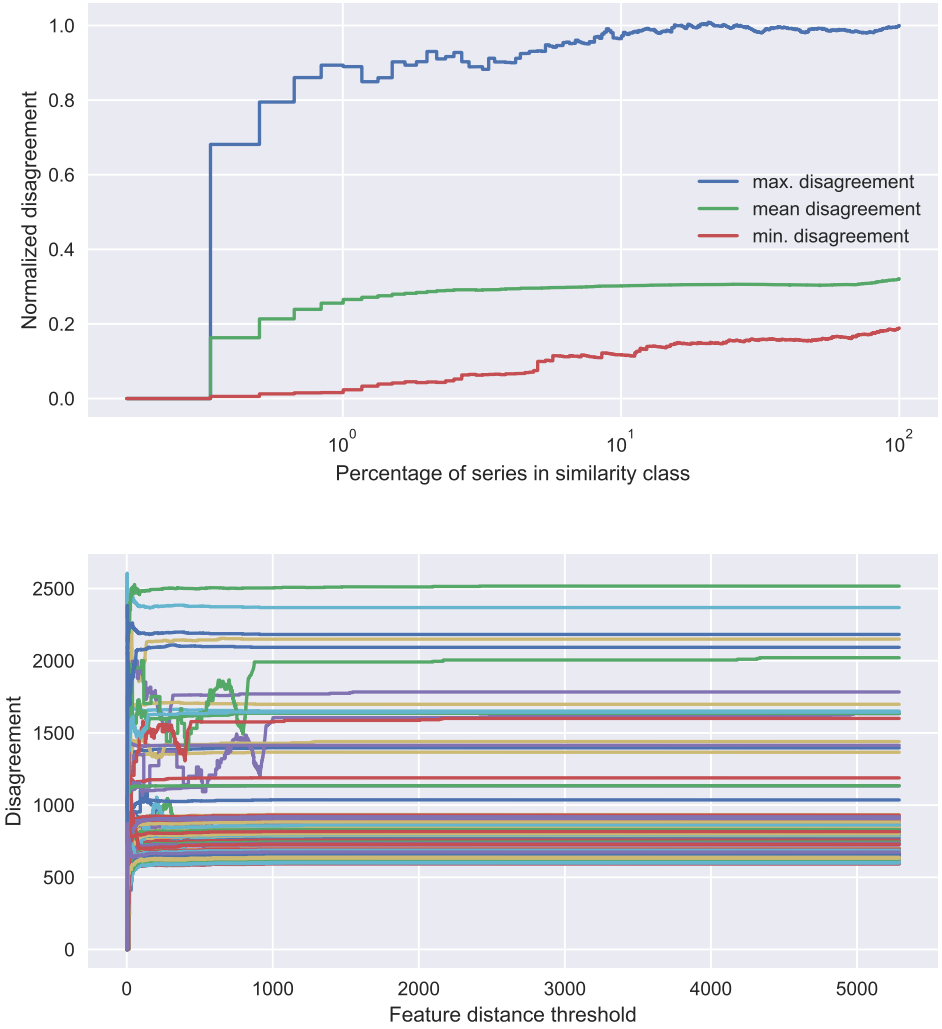

Figure 5.4: Above: example summary disagreement plot. Below: 100 of the 598 disagreement curves that generated it.

5.7 Utility evaluation

The simplest way to characterize utility is as the inverse of the distortion added to each series. This has been paired with the re-evaluation of a baseline from the benchmark [16].

Evaluating a model requires selecting and transforming the relevant data and divid-

ing it into a training set, used to learn model parameters, and a test set to gauge the model's generalization performance. Depending on the scenario, the interaction between anonymization and data mining plays out in different ways. Firstly, we need to differentiate between data that is released with a specific data analysis or prediction task in mind (*task-specific release*), versus the case in which data is released without complete knowledge of its future use (*task-agnostic release*). A distinction must also be made on what portion of the data is de-identified:

1. **Original training and test data**: this is the modality that applies to MIMIC-III. Time series are released as they were recorded and distributed semi-publicly, so that they can easily be used for research. This is a task-agnostic setting.

2. **Original training data, de-identified test set**: for instance, the case of a company training a model on proprietary data, and then selling predictions after having received de-identified data.

3. **De-identified training data, original test set**: this could for example be the case of a third-party company being employed by medical institutions to build a model. Healthcare providers would need to de-identify the data before handing it to the third-party, and after receiving the model would be able to run it on original data and verify its effectiveness.

4. **De-identified training and test data**: in the task-agnostic case, this is like (1) but with additional care to protect privacy. In the task-specific case, this setting could be produced by the intersection of (2) and (3), i.e.: a party is sold de-identified data, builds a model, accepts de-identified data and offers a prediction.

The task-agnostic setting is particularly problematic because a lot of potentially identifying information would need to be distributed so as to be able to generically filter out all the irrelevant data (which is an important requirement, as highlighted in Section 2.1), and obtain target labels. Reconstructing approximate filters and labels through indirect means or using incomplete data might train a worse model, but is especially problematic with respect to the test set, because it might change the conditions of the task. It might make the task simpler, thus driving up accuracy, but this statistic would then be dangerously misleading. As a concrete example, the "in-hospital mortality" prediction task is defined as predicting correctly whether a patient died in the ICU based on measurements recorded within the first two days of care. After anonymizing length of stays and resampling, it is not possible to know how long into a patient's stay any recording was produced. This could be estimated assuming the first recording for every patient was taken at time 0, but for some patients, recordings will be taken only later and much closer to their deaths, trivializing the prediction. Additionally, whether the patient died is inferred by whether their decease date is recorded before their discharge (information that cannot be found by looking at the time series exclusively). For these reasons, this report focuses on task-specific releases, where the data owners and model builders collaborate throughout the process. It is additionally assumed that the filtering and target labels are not identifying. In practice, this assumption should be carefully reconsidered. As a consequence, only settings (2), (3), and the task-specific scenario of (4) have been looked into. The model evaluation procedure is summarized by Figure

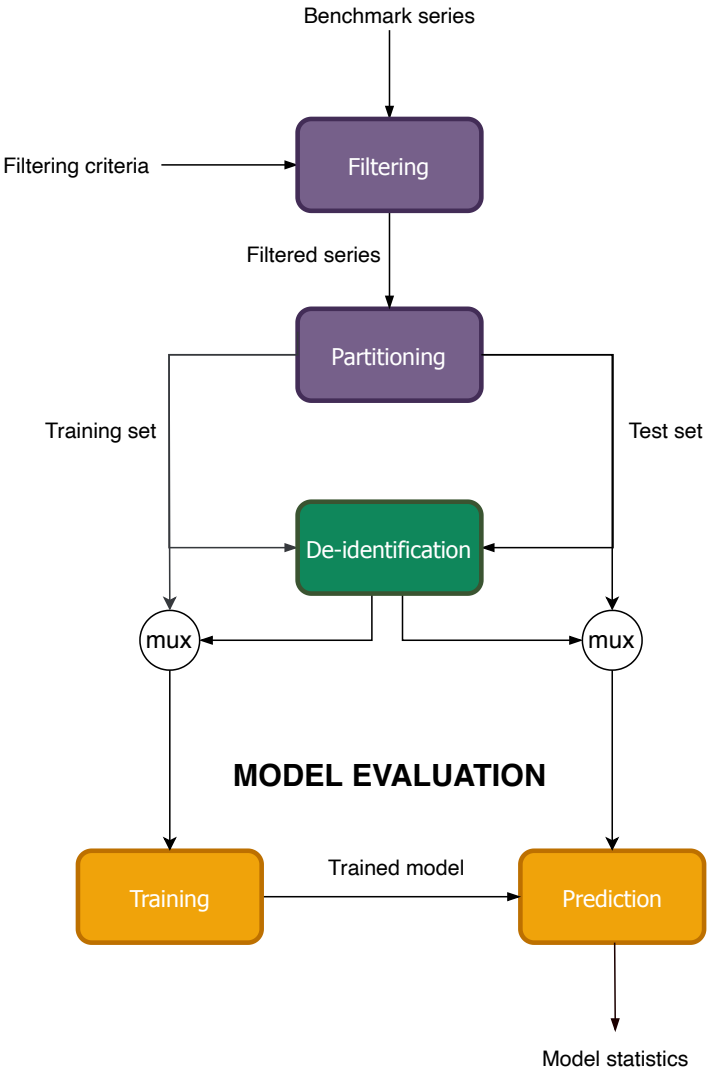

Figure 5.5: Overview of model evaluation. The circles marked with the word "MUX" are signal selectors, calibrated to match the different de-identification settings (whether training or test data should bypass de-identification).

5.5.

For settings (2) and (3), there is a mismatch between the data generation processes applied to the training and test set. This could make the test-time accuracy worse. In any case, it is paramount to verify that whenever the accuracy of a model is reported, the transformations that the evaluation data undergoes are as similar as possible to those that the model would encounter in real life. When the model builder needs to tune hyper-parameters or choose among models, that should be done with a validation set. In settings (2) and (4), this could be compiled by the data owners. In (3), the model builder would have to resort to data "stolen" from the training set.

Among the benchmark tasks listed in Section 2.1, the in-hospital mortality prediction has been analyzed. Out of the several baselines that the benchmark provides, I have utilized the simplest, logistic regression. The goal of this effort is not to achieve

state-of-the-art accuracy, but to use a baseline as a proxy for the quantification of the post-anonymization data degradation. Logistic regression is only slightly worse than the deep learning alternatives, and significantly faster to train, allowing for fast iterations between experiments. The features fed to logistic regression are, for 7 subsequences of each series, the minimum, maximum, mean, standard deviation, skew, and number of measurements. The 7 subsequences are the full time series, the first and the last 10%, 25% and 50%.

Consistently with the benchmark evaluation, I split series into training (70% of the series), validation (15% of the series) and test sets (15% of the series). I shuffled all series, deterministically across runs, by re-seeding a random generator.

Chapter 6

Results

6.1 De-identifying lengths of stay

As a preliminary exercise, I inspected the pre-processed series coming from the benchmark script to establish how identifying lengths of stay can be. It was computed as the difference between the values of the relative recording time[1] at the latest and first datapoint. After rounding durations to the nearest hour, and even before using any other known feature, it was possible to narrow down series to duration classes not larger than five elements for 489 (or approximately 1.17%) of the 41,901 series. For 220 individuals, it was possible to be uniquely identified based on their series duration, rounded to the nearest hour. This is problematic because an attacker could gather duration information through the geolocation of a mobile device belonging to a patient, i.e. it could be seen using GPS tracking when he/she enters and leaves the hospital. It is a widely known fact that many apps (legitimately or illegitimately) silently query the phone's location services, and each of them increases the attack surface. Additionally, all the identifying durations mentioned are very long, namely above 500 hours. Among them, the average distance between consecutive durations is over 5 hours. With durations so distinctive, techniques more traditional than sophisticated location data breaches could be employed. With the even coarser resolution of one day increments, it is still possible to almost identify a small number of individuals, as illustrated in Figure 6.1, allowing to exploit newspaper headlines as mentioned in Section 2.2. This motivated the duration de-identification algorithm described in Section 3.

The algorithm turned out to be moderately effective. Time indices were discarded, and thanks to resampling the resulting series could be interpreted as having one record each hour. The algorithm made sure that each duration class (granularity of one hour), had size at least k, and in doing so it allowed the retention of all series, even though a small number of them were significantly shorter. With duration $k = 50$, on average 5.93 records were lost per series (standard deviation =21.13), corresponding to 5.98% of records on average (standard deviation =17.65%). For almost the first 70% series, ordered by number of records (all with 26 records or less), all their records could be

[1] "relative recording time" refer to times whose value is the difference from the time of admission

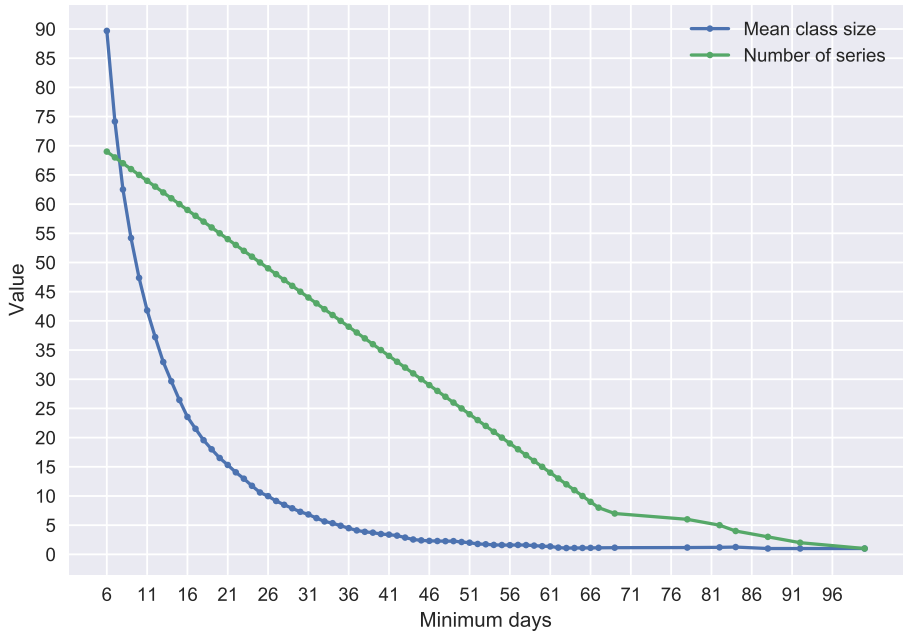

Figure 6.1: On the *x* axis, the minimum number of days of treatment. On the *y* axis, the mean size of classes based on the number of days of treatment, and the number of total series above the minimum days threshold.

kept. Suppression becomes necessary after that threshold, because as already seen, lengths become considerably sparser. With duration $k = 100$, the average records lost per series increases to 6.78 (standard deviation =24.15), which correspond to 8.67% on average (standard deviation =20.54%).

6.2 Tuning the constraint optimization

In most runs of the genetic algorithm, the computation of fitness based on annealing led to numerical overflow. This was because a large number of iterations was needed in order to find a solution that brought series reasonably close to their target feature values. Overflow meant that all solutions had fitness equal to ∞, and thus the algorithm was not able to differentiate between any of them and was thus led astray, towards solutions that added a lot of distortion while not getting close to the feature targets. To mitigate this, three approaches have been tried:

- *DF fitness*: dampen the feasibility value in the annealing formula, by dividing it by a constant (100 was chosen)
- *CI fitness*: cap the iteration value in the annealing formula to 300
- *NA fitness*: replace annealing by a weighted combination of feasibility, where feasibility is given a large weight (100), and the low-distortion objective

In order to maximize the performance of the algorithm it was necessary to choose among these three approaches and tune the other parameters. To do so, I have selected 100 series at random from 5 duration classes (20 series per class), in such a way that none of the series had missing values. Mondrian was run on the full dataset (with k=5), looking at the dimensions from Section 5.1, excluding height and weight. The genetic algorithm was configured to satisfy the constraints coming from Mondrian for each of the 5 dimensions of the 100 validation series. The performance of the algorithm was judged on the basis of two measures. The first, *average cumulative unfeasibility*, was computed as the sum of the unfeasibility of each feature value, averaged across series and dimensions. In turn, the unfeasibility for a feature is the absolute value of the difference between target and actual feature values. Distortion for a series was instead defined to be the sum of the absolute values of the additive transformations at each record. The second measure is then *average cumulative distortion*, or the distortion averaged across series, and normalized to account for different durations.

I have also looked at the performance of the ad-hoc constraint optimization algorithm, run for 50 iterations, whose results are summarized in Table 6.1. For this algorithm, it is consistently more challenging to achieve feasibility for short series than for longer ones, because single maxima and minima have a larger impact on the mean and variance, so that it is harder to "juggle" these conflicting requirements. Note that the ad-hoc algorithm will always satisfy the mean and variance targets, because they are adjusted as a final step on every iteration.

Duration	Avg. cum. unfeasibility	Avg. cum. distortion
5	1.36	10.92
25	0.50	9.86
50	0.23	9.10
75	0.16	8.98
100	0.16	9.26
All classes	0.48	9.62

Table 6.1: Results of running the ad-hoc constraint optimization on validation series, for different duration classes.

Duration	Avg. cum. unfeasibility	Avg. cum. distortion	Avg. iterations
5	1.94	11.01	1245.34
25	1.70	10.15	1369.37
50	2.37	9.40	1574.23
75	1.19	9.31	2174.65
100	3.42	7.17	2278.37
All classes	2.13	9.41	1728.39

Table 6.2: Results of running the genetic algorithm for constraint optimization on validation series with the best parameters found, for different duration classes.

The parameters that were explored for the genetic algorithm are the standard deviation

of gene mutation (2, 3 or 5), the probability of a gene mutating (0.1, 0.2 or 0.4), the tournament size (5, 10 or 15), and the number of mating parents (2, 3 or 4). The parameters found to minimize unfeasibility and distortion were 3, 0.2 and 5, respectively. The standard deviation of the initialization values was always set to 1, to slightly push initial solutions away from the null series, while initially biasing the algorithm towards solutions with low distortion. At each iteration, 24 solutions were kept, of which 2 for each chosen parent, 2 for their mutated combination, while the remaining slots were for the best past solutions.

The genetic algorithm was said to converge, and stopped, if no improvement in fitness was obtained for 300 consecutive iterations. With the NA fitness, the algorithm converged really slowly, after more than 1700 iterations on average, against less than 500 for both DF and CI. However, NA also produced the least unfeasible solutions, scoring an average cumulative unfeasibility of 2.13 with best parameters (see Table 6.2), which is worse than the output of the ad-hoc algorithm. Conversely, the genetic algorithm led to a slightly lower distortion on average (9.41). The genetic algorithm, unlike the ad-hoc one, does not have significantly worse performance than its average when run on short series. The average unfeasibility for the first four classes was always close to 2. Instead, this algorithm was slightly less effective (unfeasibility larger than 4 across most runs) with series of length 100, because of the increased search space. Similarly, the number of iterations before convergence increased with the length of series.

It must be pointed out that with the genetic algorithm, the variance target was met more precisely than the other targets in almost all cases. This is because the scale of the typical variance value is much larger than that of the other features, so that fitness incurs greater penalty, and the algorithm is more decisively encouraged to concentrate on variance. This could, in theory, be addressed by normalizing feature values and targets. Sometimes the prioritization of variance resulted in a distortion that significantly altered the original series (Figure 6.2). More frequently, when the target variance was not too distant, it was possible to preserve the shape of the original series (Figure 6.3). To decouple the performance of the genetic algorithm from that of feature de-identification with Mondrian, the next section makes use of the ad-hoc algorithm.

6.3 Feature de-identification

Comparing disagreement pre and post de-identification is challenging because the effects of clustering series by feature and distorting series are entangled. When disagreement on a series x is higher for low distance thresholds than in the original data, is it because more series have been pushed close to x, or because its features have been heavily distorted so that they lie far from the truth (corresponding to the adversarial expectation)? In practice, it is going to be due to a combination of these two factors. For this reason, by looking at disagreement alone, it is not possible to decide which of the range heuristic or diversity heuristic perform better. Even though the latter should intuitively produce more disagreement, it might do so principally because it is less economical when partitioning the feature space, and thus increases disagreement mostly

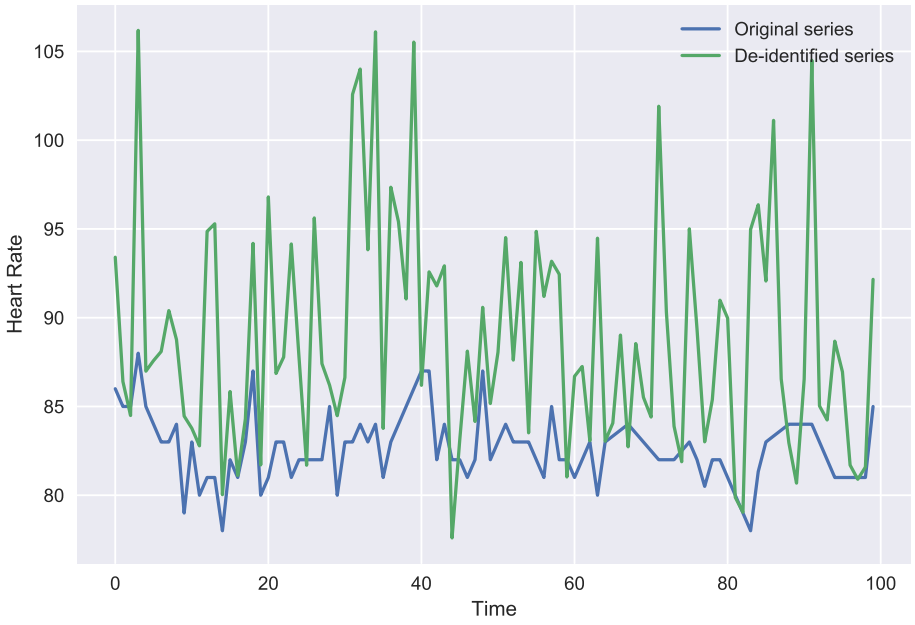

Figure 6.2: Example of genetic algorithm producing an output with high distortion. Given features *minimum*, *maximum*, *mean*, *variance*, the original feature values were (78, 88, 82.66, 3.55), the targets were (77.75, 106.19, 90.20, 41.52), and the result feature values were (77.60, 106.18, 88.94, 41.52)

by distortion. That is undesirable: in an extreme case, a simple algorithm to trivially increase disagreement is to add a lot of random noise to all series, and make their released versions disagree with the truth. In fact, after having run Mondrian with the two heuristics on 15 duration classes (ranging from length 3 to 500, and corresponding to almost 10% of the full data), in all cases the feature targets obtained with the diversity heuristic demanded a larger distortion. I thus chose to utilize the range heuristic in all successive trials.

Tuning the other parameters is similarly complicated. Just in the same way Δ-AUC is an incomplete statistic for measuring the privacy of a dataset, it is inadequate to compare privacy gains (or losses), because it gives too much weight to the right-most points of the curve. This happens because those points are going to have the most disagreement, while the most interesting changes indeed happen in the left-most section of the curve. Preliminary runs of the pipeline show that often Δ-AUC decreased, even in the face of a significant improvement for low thresholds, and this was due to lower disagreement at the largest distance thresholds. This lower disagreement means that the resulting data release has series which are all a bit more similar to each other. This is an expected result of passing feature values through Mondrian, which more heavily impacts outliers by bringing them closer to more common values. I used the 15 duration classes from the previous experiments to compare the feature k values of 5 and 10, and manually inspected each result. These k values were chosen so that they were not too large with

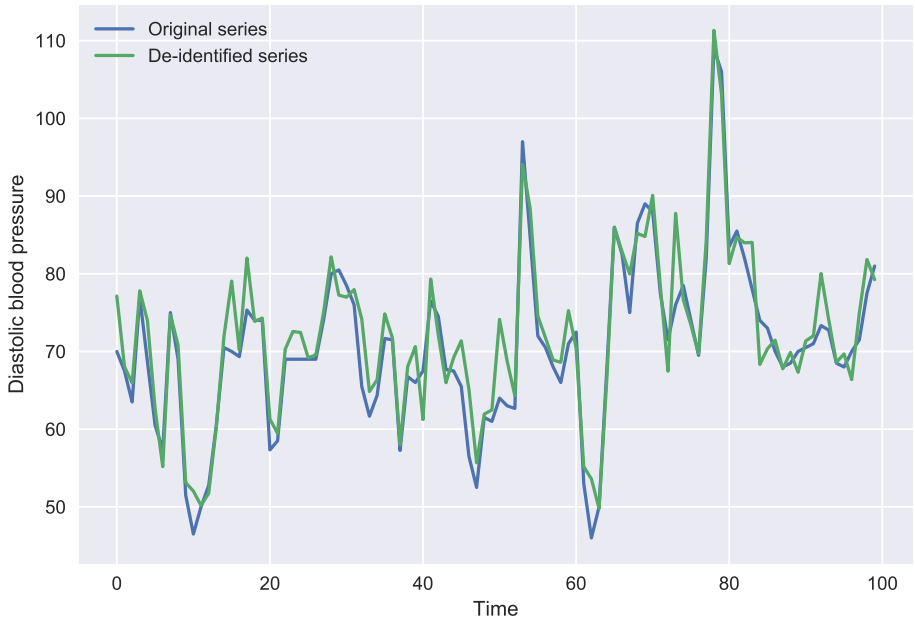

Figure 6.3: Example of genetic algorithm producing a good output, with low distortion. Given features *minimum*, *maximum*, *mean*, *variance*, the original feature values were (46, 109, 70.64.66, 115.61), the targets were (49.83, 111.31, 72.18, 108.90), and the result feature values were (49.78, 111.33, 72.14, 108.92)

respect to the smallest duration classes (of size 100). In all cases, increasing feature k led both to lower Δ-AUC and larger distortion. The reason for the Δ-AUC decrease was again the lower disagreement at the largest thresholds. As for distortion, larger equivalence classes mean larger average feature distance from the corresponding feature target. For the sake of illustrating the effect of de-identification and assessing utility, $k = 10$ was chosen, which consistently led to better disagreement at low thresholds.

Figure 6.4 represents the mean curve obtained for the duration class of 30-records series. It shows disagreement being higher for small inclusion percentages, then the two curves intersecting at middle inclusion percentages, and finally disagreement becoming lower after de-identification. These characteristics were common to all the maximum and mean disagreement curves plotted, and to some of the minimum disagreement curves. Unfortunately, in a minority of cases, the minimum curves only showed an improvement for large inclusion percentages, or exhibited a more erratic behavior, as the plot for the length 9 class in Figure 6.5 shows.

These results give an idea of the shape of disagreement curves and how it changes. It remains to be seen what concrete effects this change has on potential attacks. To explore this, consider an adversary that takes similarity classes with distance threshold as small as needed to gather at least one series. Before de-identification, the reconstruction error

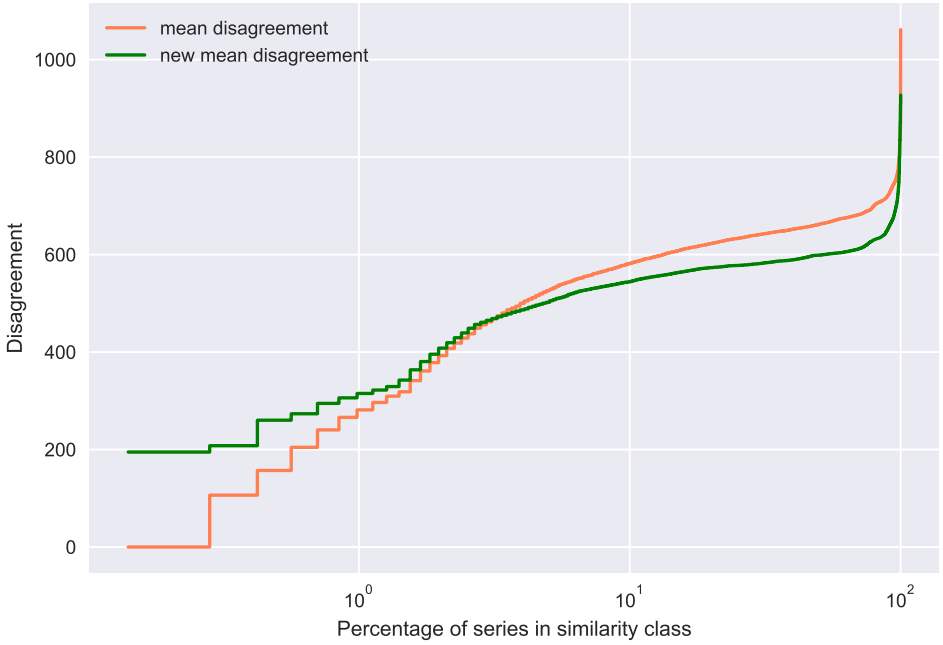

Figure 6.4: Example mean disagreement curve after feature de-identification.

is always 0. After de-identification, the reconstruction error would be high on average. For example, with the length 9 class, whose curve was analyzed before, for more than 85% of all series, disagreement was at least 35. It is worth reminding the reader that 35 is the Euclidean distance across all records, normalized by the the number of records. By manipulating the formula, it is possible to see that if the difference was homogenous across records, a disagreement of 35 means a per-record disagreement of 35 times the square root of the number of records. Even ignoring the term under the square root, for most units of measurement taken into consideration, 35 is a significant difference. On the other end of the spectrum, if the disagreement was exclusively in one record, the entity of the difference for that record would be 35 times the number of records. What about the remaining 15%? Most of it still has disagreement equal to 0. This is due to failures in the achievement of target features and to the original series still being the most similar regardless of the additive distortion. A data owner might consider suppressing entirely such series, depending on the situation. Luckily, for at least a few of those problematic series (albeit only for a minority), the minimum feature distance threshold necessary to include them increased (from its initial value of 0), adding some uncertainty detrimental for realistic attackers. Additionally, the attacker could reasonably use higher thresholds and include more series in the reconstruction class. In this case, the disagreement of the de-identified series would be better as argued before. On the length 30 class, also discussed previously, the results were even better, and are summarized in Figure 6.6. Similar results were obtained on all other duration classes.

One of the most tricky aspects of this framework is the identification of sensitive

Figure 6.5: Example of minimum disagreement curve after feature de-identification. Improvements are obtained only close to a 10% inclusion percentage. The irregularity of the new curve is due to the minimum function, selecting series for which disagreement has not changed much.

features. To highlight this problem I have de-identified once again, but passing different features to Mondrian and to the evaluation of disagreement. Specifically, I stopped looking at variances for the former. The result of this operation is unsurprisingly the decrease of the degree of protection, but not the renunciation of all privacy gains that the algorithm unlocked, given the new threat model. For example, the curve in Figure 6.6 was replotted, and the shape of the result was very similar, but with the inflection point at about $10^{1.5} \approx 31.6$, rather than at more than 10^2. For the length 9 duration class, the number of series whose first disagreement value is lower than 35 doubled from 15 to about 30%.

6.4 Trust settings and model performance

When using the ad-hoc constraint optimization algorithm, the results summarized in Tables 6.3 and 6.4 were obtained. Unsurprisingly, the best training set results are obtained when not de-identifying it. This is probably because the de-identified dataset will present more inconsistencies that the model needs to work around. It appears that the best test results are obtained when de-identification is applied to both the training and test sets, by a significant margin. In particular, the model exhibits very poor accuracy when only de-identifying training data. This is perhaps the consequence of the model's attempt to adjust to the idiosyncrasies of de-identification, which is a bad

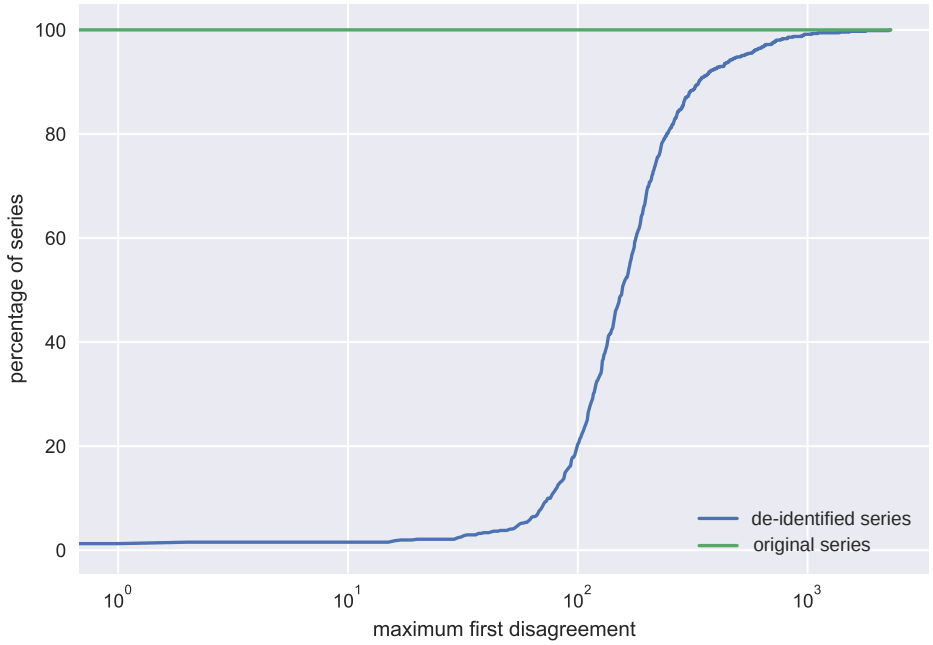

Figure 6.6: For the duration class in which series have 30 records, before and after feature de-identification, the percentage of series whose *first disagreement value* was not larger than the values on the x axis (on a logarithmic scale). The green line for original series means that for all of them a precise expectation of the feature values would allow a complete reconstruction. The blue line is a considerable improvement on that, as a large number of series have large first disagreement values.

Training / test	orig. / orig.	de-id. / de-id.	orig. / de-id.	de-id. / orig.
ROC-AUC	0.87	0.87	0.86	0.64
PR-AUC	0.54	0.54	0.51	0.22

Table 6.3: Results of in-hospital mortality prediction on the training set given different trust settings

Training / test	orig. / orig.	de-id. / de-id.	orig. / de-id.	de-id. / orig.
ROC-AUC	0.85	0.83	0.82	0.64
PR-AUC	0.49	0.45	0.40	0.22

Table 6.4: Results of in-hospital mortality prediction on the test set given different trust settings

strategy if the test set does not present the same characteristics.

The setting in which both datasets are de-identified was also investigated when using the genetic algorithm. In this case, the performance of the model was poorer, with the same *ROC-AUC* value but with a *PR-AUC* equal to 0.43 (a 0.02 decrease).

Chapter 7

Conclusion

7.1 Discussion of experiments

This report has highlighted some worrying results on the re-identification potential of the series included in the MIMIC-III dataset. As long as feature-based attacks are realistic, and features available to the attacker can be at least in part estimated, it was shown that under a reasonable threat model, the algorithm based on Mondrian and constraint optimization provides effective mitigation. This algorithm is, to my knowledge, an entirely novel approach to the de-identification of time series, and allows for minimum data distortion, whose extent is constant (rather than exponential) in the length of source series. Unlike other frameworks, such as differential privacy, this approach requires a large number of assumptions on the adversarial background knowledge. However, it can potentially require far fewer assumptions on the final application of the data, and is based on the full release of the data (though distorted), rather than on the release of a summary. It could be argued that, in the absence of a single point-estimate of privacy, my approach is more problematic to use. It must be kept in mind, however, that even differential privacy needs an expert determination of what the appropriate parameters (ϵ and δ) are, and there is a general lack of understanding of what different parameter settings mean for concrete privacy under various conditions [6]. Nevertheless, pointing out that my approach does not provide any theoretically-provable guarantees, but only practical improvements, is a valid criticism.

The concept of disagreement was a useful tool in the analysis of the de-identification algorithm, and it could also be used to quantify the potential for re-identification in a novel way. The duration de-identification component was also a useful innovation, acting on a source of identifiability that is definitely under-studied.

The impact of de-identification on the performance of in-hospital mortality was undeniable, but only limited, and reasonably encouraging. The most noteworthy result is the necessity of de-identifying both data used for training and at test-time, against the intuition that having as much original data in the training and evaluation would benefit performance. This seems to suggest that the model learns to adapt to de-identification, and expects test data to exhibit the same occasional inconsistencies.

The genetic algorithm was complicated to tune and the slowest component to run. It was shown that in some circumstances it is possible to design an ad-hoc algorithm to replace it, with gains both in terms of runtime and privacy. However, the main advantage of the genetic algorithm is its generality. It must be kept in mind that it is a non-essential component, and other generic optimization algorithms could be adapted to replace it. The random component of the genetic algorithm, which could be useful against attacks that reverse-engineer deterministic de-identification algorithms, could also be incorporated in other algorithms.

7.2 Future work

There are numerous directions for expanding on the ideas presented in this report. The techniques discussed are only a first step and should be refined. The genetic algorithm should be optimized further, perhaps by restricting the space of solutions, leveraging domain-specific knowledge. More advanced combinations of features should be addressed, and this should include some more sophisticated pattern-matching attacks. The concept of disagreement should be explored thoroughly so that indications can be provided to data owners regarding the implications of different privacy-levels. In this way a consensus could be built on reasonable values of disagreement given different contexts. Refining the concept of disagreement includes finding appropriate ways to systematically aggregate the disagreement of all the series in a data release, rather than looking at individual curves. It would be interesting to study more in depth disagreement and to what extent its cause is the additive noise rather than the clustering.

Attacks on the method discussed should be explored; in particular it should be seen whether it is possible to filter out some of the uncertainty introduced by the additive noise by leveraging biases in the anonymization process. Additionally, realistic attacks should be investigated, based on databases of medical data as models of background knowledge. It is also interesting to look at how the model changes to adapt to de-identified data, together with ways that could be exploited by adversaries.

It would be interesting to see how the notions discussed here can be combined with traditional k-anonymity, for the combined release of rectangular data (such as demographics) and time series. Finally, an important area of research is devising strategies to apply feature-based de-identification with unspecified or partially-specified applications in mind (in terms of labelling and filtering), and identifying the characteristics of the trade-off between (lack of) specification and data loss.

Bibliography

[1] Charu C. Aggarwal. On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st International Conference on Very Large Data Bases*, pages 901–909. VLDB Endowment, 2005.

[2] Arthur Asuncion and David Newman. UCI repository of machine learning databases, 1992, 2007.

[3] Susan E. Carlson and Ronald Shonkwiler. Annealing a genetic algorithm over constraints. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 4, pages 3931–3936. IEEE, 1998.

[4] S.F. Clarke and J.R. Foster. A history of blood glucose meters and their role in self-monitoring of diabetes mellitus. *British journal of biomedical science*, 69(2):83–93, 2012.

[5] Francis S. Collins and Lawrence A. Tabak. Policy: NIH plans to enhance reproducibility. *Nature News*, 505(7485):612, 2014.

[6] Fida Kamal Dankar and Khaled El Emam. Practicing differential privacy in health care: A review. *Trans. Data Privacy*, 6(1):35–67, 2013.

[7] Thomas Desautels, Jacob Calvert, Jana Hoffman, Melissa Jay, Yaniv Kerem, Lisa Shieh, David Shimabukuro, Uli Chettipally, Mitchell D. Feldman, Chris Barton, et al. Prediction of sepsis in the intensive care unit with minimal Electronic Health Record data: a machine learning approach. *JMIR medical informatics*, 4(3), 2016.

[8] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[9] Khaled El Emam, Fida Kamal Dankar, Romeo Issa, Elizabeth Jonker, Daniel Amyot, Elise Cogo, Jean-Pierre Corriveau, Mark Walker, Sadrul Chowdhury, Regis Vaillancourt, et al. A globally optimal k-anonymity method for the de-identification of health data. *Journal of the American Medical Informatics Association*, 16(5):670–682, 2009.

[10] M.J. Elliot, C. Dibben, H. Gowans, E. Mackey, D. Lightfoot, K. O'Hara, and K. Purdam. Functional anonymisation: The crucial role of the data environment

in determining the classification of data as (non-) personal. *CMIST work paper*, 2, 2015.

[11] Amparo Gil, Javier Segura, and Nico M. Temme. *Numerical methods for special functions*, volume 99. Siam, 2007.

[12] Aris Gkoulalas-Divanis and Vassilios S Verykios. Concealing the position of individuals in location-based services. *Operational Research*, 11(2):201–214, 2011.

[13] Jeffrey E. Gotts and Michael A. Matthay. Sepsis: pathophysiology and clinical management. *Bmj*, 353:i1585, 2016.

[14] Ulrich Greveler, Benjamin Justus, and Dennis Loehr. Multimedia content identification through smart meter power usage profiles. *Computers, Privacy and Data Protection*, 1(10), 2012.

[15] Marian Haescher, Denys JC Matthies, John Trimpop, and Bodo Urban. A study on measuring heart-and respiration-rate via wrist-worn accelerometer-based seismocardiography (SCG) in comparison to commonly applied technologies. In *Proceedings of the 2nd international Workshop on Sensor-based Activity Recognition and Interaction*, pages 1–6, 2015.

[16] Hrayr Harutyunyan, Hrant Khachatrian, David C. Kale, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *arXiv preprint arXiv:1703.07771*, 2017.

[17] Baik Hoh and Marco Gruteser. Protecting location privacy through path confusion. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 194–205. IEEE, 2005.

[18] Alistair E.W. Johnson, Tom J. Pollard, Lu Shen, H. Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G. Mark. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3, 2016.

[19] Stephan Kessler, Erik Buchmann, and Klemens Böhm. Deploying and evaluating pufferfish privacy for smart meter data. In *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*, pages 229–238. IEEE, 2015.

[20] Matthieu Komorowski, Leo A. Celi, Omar Badawi, Anthony C. Gordon, and A. Aldo Faisal. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature Medicine*, 24(11):1716–1720, 2018.

[21] David Kotz, Carl A. Gunter, Santosh Kumar, and Jonathan P. Weiner. Privacy and security in mobile health: a research agenda. *Computer*, 49(6):22–30, 2016.

[22] Fabian Laforet, Erik Buchmann, and Klemens Böhm. Individual privacy constraints on time-series data. *Information Systems*, 54:74–91, 2015.

[23] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 49–60. ACM, 2005.

[24] Kristen LeFevre, David J. DeWitt, and Raghu Ramakrishnan. Mondrian multidimensional k-anonymity. In *Proceedings of the 22nd International Conference on Data Engineering*, pages 25–25. IEEE, 2006.

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

[26] 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. ACM, 2012.

[27] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

[28] David M. Maslove, Francois Lamontagne, John C. Marshall, and Daren K. Heyland. A path to precision in the ICU. *Critical Care*, 21(1):79, 2017.

[29] Leonardo Mazzone. Database de-identification for electronic health records, 2019.

[30] Brad L. Miller, David E. Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212, 1995.

[31] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[32] Andrés Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, and David Irwin. Private memoirs of a smart meter. In *Proceedings of the 2nd ACM workshop on embedded sensing systems for energy-efficiency in building*, pages 61–66. ACM, 2010.

[33] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy*, pages 111–125. IEEE, 2008.

[34] Spiros Papadimitriou, Feifei Li, George Kollios, and Philip S Yu. Time series compressibility and privacy. In *Proceedings of the 33rd international conference on Very large data bases*, pages 459–470. VLDB Endowment, 2007.

[35] Niranjani Prasad, Li-Fang Cheng, Corey Chivers, Michael Draugelis, and Barbara E. Engelhardt. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. *arXiv preprint arXiv:1704.06300*, 2017.

[36] Reza Rawassizadeh, Blaine A. Price, and Marian Petre. Wearables: Has the age of smartwatches finally arrived? *Communications of the ACM*, 58(1):45–47, 2014.

[37] Stuart J. Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Pearson Education Limited, 2016.

[38] Dymitr Ruta, Ling Cen, and Ernesto Damiani. Fast summarization and anonymization of multivariate big time series. In *2015 IEEE International Conference on Big Data (Big Data)*, pages 1901–1904. IEEE, 2015.

[39] Pierangela Samarati. Protecting respondents identities in microdata release. *IEEE transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[40] Joan Serra and Josep Ll. Arcos. An empirical evaluation of similarity measures for time series classification. *Knowledge-Based Systems*, 67:305–314, 2014.

[41] Benjamin Shickel, Patrick James Tighe, Azra Bihorac, and Parisa Rashidi. Deep EHR: A survey of recent advances in deep learning techniques for electronic health record (EHR) analysis. *IEEE journal of biomedical and health informatics*, 22(5):1589–1604, 2018.

[42] Reza Shokri, George Theodorakopoulos, Jean-Yves Le Boudec, and Jean-Pierre Hubaux. Quantifying location privacy. In *2011 IEEE symposium on security and privacy*, pages 247–262. IEEE, 2011.

[43] Lidan Shou, Xuan Shang, Ke Chen, Gang Chen, and Chao Zhang. Supporting pattern-preserving anonymization for time-series data. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):877–892, 2011.

[44] Huan Song, Deepta Rajan, Jayaraman J. Thiagarajan, and Andreas Spanias. Attend and diagnose: Clinical time series analysis using attention models. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[45] Latanya Sweeney. Datafly: A system for providing anonymity in medical data. In *Database Security XI*, pages 356–381. Springer, 1998.

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

[47] Latanya Sweeney. Only you, your doctor, and many others may know. *Technology Science*, 2015092903(9):29, 2015.

[48] Nazanin Takbiri, Amir Houmansadr, Dennis L Goeckel, and Hossein Pishro-Nik. Matching anonymized and obfuscated time series to users' profiles. *IEEE Transactions on Information Theory*, 65(2):724–741, 2018.

[49] Hao Wang and Zhengquan Xu. CTS-DP: Publishing correlated time-series data via differential privacy. *Knowledge-Based Systems*, 122:167–179, 2017.

[50] Leon Willenborg and Ton De Waal. *Elements of statistical disclosure control*, volume 155, page 27. Springer Science & Business Media, 2012.

[51] Mohammad-Reza Zare-Mirakabad, Fatemeh Kaveh-Yazdy, and Mohammad Tahmasebi. Privacy preservation by k-anonymizing ngrams of time series. In *2013*

10th International ISC Conference on Information Security and Cryptology (ISCISC), pages 1–6. IEEE, 2013.

[52] Gaofeng Zhang, Xiao Liu, and Yun Yang. Time-series pattern based effective noise generation for privacy protection on cloud. *IEEE Transactions on Computers*, 64(5):1456–1469, 2014.

[53] Wei Zhou and Selwyn Piramuthu. Security/privacy of wearable fitness tracking IoT devices. In *2014 9th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–5. IEEE, 2014.

[54] Ye Zhu, Yongjian Fu, and Huirong Fu. On privacy in time series data mining. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 479–493. Springer, 2008.

[55] Ye Zhu, Yongjian Fu, and Huirong Fu. Preserving privacy in time series data classification by discretization. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 53–67. Springer, 2009.

Appendix A

Summary of notation

Symbol	Explanation
$\mathcal{D}$	universe of possible databases
$\mathcal{X}$	universe of possible series
d	original database
d^*	knowledge base of an adversary
x	original time series
q	number of dimensions in a database
$x_j^{(i)}$	record of x at index i and dimension j
$\tilde{d}$	released database
$\tilde{x}$	released series
$\hat{x}$	reconstructed time series
x^*	series used by adversary to estimate features of x
$\bar{x}$	additive noise on series x
τ_x	mapping between sequence and time indices for series x
T^x	sequence of time indices for series x
k	value for k-anonymity
$\mathcal{A}$	universe of constrained adversaries
a	adversary
$\mathcal{P}_d$	set of all identities in database d
P^x	identity associated to series x
$\delta(y,z)$	distance between series y and z
$\Delta(x,S)$	*disagreement* with series x for all $x' \in S$
$\mathcal{F}$	universe of possible features
$\mathcal{F}^*$	features used in attack
$\approx_t^*$	similarity operator under distance threshold t
$C_{\mathcal{F}^*,d,t}(x)$	similarity class for $x \in d$ at threshold t, given feature set $\mathcal{F}^*$

Appendix B

Data management plan

This project was completed as part of the Master of Informatics degree program at the University of Edinburgh. In order to comply with the guidelines of the School of Informatics and in the presence of sensitive data being handled, a data management plan has been completed by filling a template through the tool available at https://dmponline.dcc.ac.uk/. This tool is principally aimed at producing statements over the collection of new data. This means that a large subset of the questions asked was not relevant, and the resulting data management plan is quite compact.

What data will be generated or reused in this research?
No original data will be generated in this research. We will be using the dataset MIMIC-III, from https://mimic.physionet.org/.

How will the data be documented to ensure it can be understood?
As no original data will be generated, we refer to the documentation at http://archive.is/vhCJh.

Where will the data be stored and backed-up?
The data will be stored on a private hard drive, and will not be backed up, as it is hosted at https://mimic.physionet.org/ and can easily be recovered from there.

How will you quality assure your data?
The data is provided with an MD5 checksum, which will be verified before the beginning of the research.

How will you manage any ethical and IPR issues?
Before obtaining the dataset, a compulsory training has been completed on ethics in health research involving human participants and on the legal framework under which the dataset was released (HIPAA). Additionally, an agreement has been signed that prevents us from attempting to re-identify the individuals in the dataset.

Which data do you plan to keep and for how long?
The entirety of the MIMIC-III dataset will be kept until June 2020.

Can you share your data? If not, how will it will be stored and preserved?
The data cannot be directly shared, and will be protected by keeping it in encrypted form on a private hard disk.