PPaaS: Privacy Preservation as a Service

Mahawaga Arachchige, Pathum Chamikara; Bertok, Peter; Khalil, Ibrahim; Liu, D; Camtepe, Seyit

Mahawaga Arachchige, P. C., Bertok, P., Khalil, I., Liu, D., & Camtepe, S. (2021). PPaaS: Privacy Preservation as a Service. Computer Communications, 173(1), 192–205. https://doi.org/10.1016/j.comcom.2021.04.006
Document Version: Accepted Manuscript

Published Version: https://doi.org/10.1016/j.comcom.2021.04.006

PPaaS: Privacy Preservation as a Service

M.A.P. Chamikara[a,b,*], P. Bertok[a], I. Khalil[a], D. Liu[b], S. Camtepe[b]

[a] *RMIT University, Australia*
[b] *CSIRO Data61, Australia*

Abstract

The published article can be found at `https://doi.org/10.1016/j.comcom.2021.04.006`

Personally identifiable information (PII) can find its way into cyberspace through various channels, and many potential sources can leak such information. Data sharing (e.g. cross-agency data sharing) for machine learning and analytics is one of the important components in data science. However, due to privacy concerns, data should be enforced with strong privacy guarantees before sharing. Different privacy-preserving approaches were developed for privacy preserving data sharing; however, identifying the best privacy-preservation approach for the privacy-preservation of a certain dataset is still a challenge. Different parameters can influence the efficacy of the process, such as the characteristics of the input dataset, the strength of the privacy-preservation approach, and the expected level of utility of the resulting dataset (on the corresponding data mining application such as classification). This paper presents a framework named Privacy Preservation as a Service (PPaaS) to reduce this complexity. The proposed method employs selective privacy preservation via data perturbation and looks at different dynamics that can influence the quality of the privacy preservation of a dataset. PPaaS includes pools of data perturbation methods, and for each application and the input dataset, PPaaS selects the most suitable data perturbation approach after rigorous evaluation. It enhances the usability of privacy-preserving methods within its pool; it is a generic platform that can be used to sanitize big data in a granular, application-specific manner by employing a suitable combination of diverse privacy-preserving algorithms to provide a proper balance between privacy and utility.

Keywords: data privacy, privacy preservation, privacy preservation as a service, data perturbation, machine learning

*Corresponding author
Email address: `pathumchamikara.mahawagaarachchige@rmit.edu.au` (M.A.P. Chamikara)

1. Introduction

Cyberspace users cannot easily avoid the possibility of their identity being incorporated in data that exposes various aspects of their lives [1]. Our day-to-day life activities are tracked by smart devices, and the unavoidable exposure of personally identifiable information (PII) such as fingerprint, facial features can lead to massive privacy loss. The heavy use of PII in social networks, in the health-care industry, and by insurance companies, in smart grids makes privacy protection of PII extremely complex. Literature shows more than a few methods to address the growing concerns related to user privacy. Among these methods, disclosure control of microdata has become widely popular in the domain of data mining [1]; it works by applying different privacy-preserving mechanisms to the data before releasing them for analysis. Privacy-preserving data mining (PPDM) applies disclosure control to data mining in order to preserve privacy while generating knowledge [1].

The main approaches to PPDM use data perturbation (modification) or encryption; literature shows a plethora of privacy preservation approaches under these two categories [2]. There has been more interest in data perturbation due to its lower complexity compared to encryption. Additive perturbation, random rotation, geometric perturbation, randomized response, random projection, microaggregation, hybrid perturbation, data condensation, data wrapping, data rounding, and data swapping are some examples of basic data perturbation algorithms, which show different behavior on different applications and datasets [3, 4, 5, 6, 7]. We can also find a number of hybrid approaches that combine basic perturbation approaches.

The availability of many privacy preservation approaches has its drawback: the selection of the optimal perturbation algorithm for a particular problem can be quite complex; Figure 1 shows different constraints that need to be considered. Different characteristics of privacy models (e.g. k-anonymity, l-diversity, t-closeness, differential privacy ([2])), different properties of privacy preservation algorithms (e.g. geometric perturbation, data condensation, randomized response), different dynamics of the input data (e.g. the statistical properties, the dimensions), and different types of applications at hand (e.g. data clustering, deep learning) are examples of the attributes that influence the effectiveness of privacy preservation and the usability of the results. At the same time, this diversity enables the selection of the privacy preservation algorithm that best suits a particular application. There is no generic approach to identify the exact levels of privacy loss vs. utility loss, given a list of privacy preservation algorithms on specific applications and datasets. Furthermore, many privacy preservation approaches fall out of favour because their applicability is not properly identified. We introduce a new approach named “Privacy Preservation as a Service ” (PPaaS) that employs a novel strategy to apply customized perturbation based on the requirements of the problem at hand and the characteristics of the input dataset.

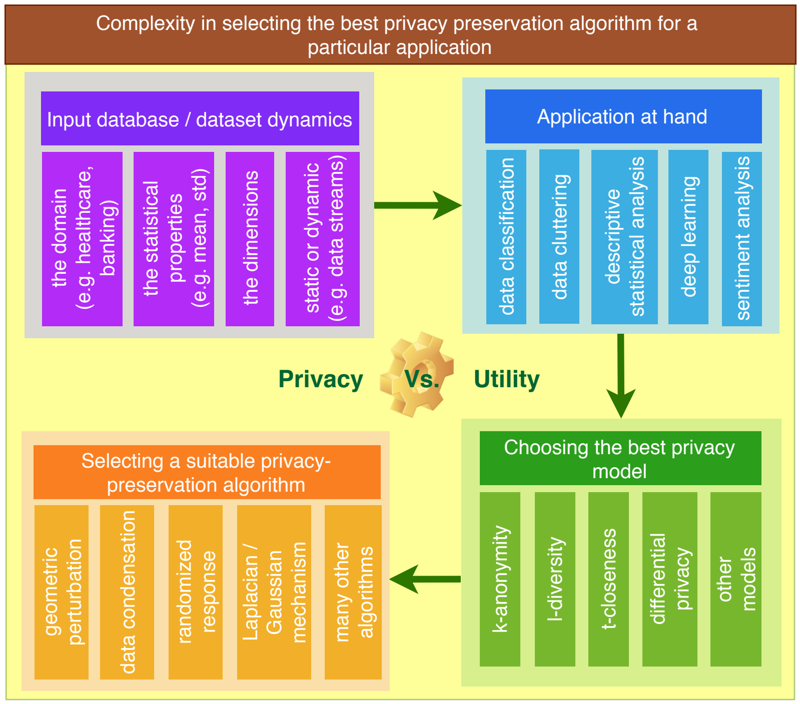

Figure 1: Complexity of selecting the best privacy preservation approach for a particular application/ database

PPaaS presents a unified service that understands data requesters’ needs and data owners’ (who have full access privileges to the raw input databases which are represented by the lowest layer Figure 4) requirements; it can facilitate privacy-preserving data sharing and can identify the best data perturbation approach. While an exhaustive analysis of all privacy protection methods for a given set of data is not feasible, a quantitative evaluation of selected relevant methods can significantly improve the efficacy of privacy protection. An appropriate set of performance and security metrics describes the quality of such a service, which is used to tailor the best privacy preservation to stakeholders’ needs. The proposed framework collects efficient privacy preservation methods into a pool and applies the approach that best suits both data owner and data requester to the data before making the data available. The selection of the best perturbed dataset is done based on attack resistance analysis integrated into privacy and utility evaluation using a fuzzy inference system (FIS).

1.1. Rationale and technical novelty

Developing generic privacy-preserving methods for data mining and statistics is challenging due to the large number of constraints that need to be considered. As the complexity of the applications increases, generic approaches often end up with low utility or low privacy ([8]). Many researchers try to overcome this by focusing on a distinct objective (e.g privacy in deep learning) ([9, 10]). As a result, there are a number of algorithms for some areas such as deep learning, with many viable privacy preservation solutions ([11]). The algorithms having unique features and characteristics, choosing the best one for a particular case can be highly complex.

PPaaS reduces the burden of choosing the optimal privacy-preserving algorithm and providing the best protection for the application and dataset at hand by introducing a unified service for the purpose. Since there can be more than one method appropriate for a particular application and dataset, empirical evaluation is utilized in this process. PPaaS manages a pool of data perturbation algorithms suitable for particular applications and a pool of potential data reconstruction attacks that can reconstruct the original data from perturbed data. When a certain application/dataset is presented, PPaaS assesses the data perturbation algorithms and produces a unified metric named fuzzy index (FI) derived from a fuzzy model. Conventionally, a fuzzy model is used to model the vagueness and impreciseness of information in a real-world problem using fuzzy sets. In PPaaS, we use a fuzzy model to select the best perturbed dataset based on privacy, attack resistance, and utility from the corresponding perturbed instances. PPaaS utilizes only robust privacy preservation approaches for data perturbation in order to avoid potential data reconstruction attacks on the perturbed data. Privacy protection is aiming at reducing the leakage of information in responses to legitimate queries. However, it was shown that the existing privacy preservation approaches are still vulnerable to different privacy attacks [12]. The attack resistance module of PPaaS is responsible for evaluating the robustness of a particular perturbed dataset against different attacks. We measure the robustness/resistance of a particular perturbed dataset against data reconstruction attacks and generate a minimum guarantee or resistance (MGR); the higher the miminum guarantee, the better the perturbation algorithm used for the perturbation of the dataset. We use quantitative definitions of utility and privacy along with MGR as inputs to the Fuzzy model. The higher the fuzzy index, the better the balance between privacy and utility under the given circumstances. The release of a particular output depends on a configurable threshold value of the corresponding FI. If the required threshold is not reached, the application of the corresponding pool is assessed repeatedly with different algorithms and parameters until one of the privacy preservation algorithms in the pool generates a satisfactory FI ($\geq$ threshold FI) for an application and dataset or the user-defined maximum number of iterations reached. With this approach, users are guaranteed to be given the best possible privacy preservation while providing optimal utility.

2. Literature

Data privacy focuses on impeding the estimation of the original data from the sanitized data, while utility concentrates on preserving application-specific properties and information ([13]). It has been noted that privacy preservation mechanisms decrease utility in general, i.e. they reduce utility to improve privacy, and finding a trade-off between privacy protection and data utility is an important issue ([14]). In fact, privacy and utility are often conflicting requirements: privacy-preserving algorithms provide privacy at the expense of utility. Privacy is often preserved by modifying or perturbing the original data, and a common way of measuring the utility of a privacy-preserving method is to investigate perturbation biases ([15]). This bias is the difference between the result of a query on the perturbed data and the result of the same query on the original data. Wilson et al. examined different data perturbation methods and identified Type A, B, C, and D biases, along with an additional bias named Data Mining (DM) bias ([15]). Type A bias occurs when the perturbation of a given attribute causes summary measures to change. Type B bias is the result of the perturbation changing the relationships between confidential attributes, while in case of Type C bias, the relationship between confidential and non-confidential attributes changes. Type D bias means that the underlying distribution of the data was affected by the sanitization process. If Type DM bias exists, data mining tools will perform less accurately on the perturbed data than they would on the original dataset.

An investigation of existing privacy preservation approaches also suggests that they often suffer from utility or privacy issues when they are considered for generic applications ([2]). Methods such as additive perturbation with noise (for differentially private data) can produce low utility due to the highly randomized nature of added noise ([16, 6]). Randomized response, another privacy preservation approach, has the same issue and produces low utility data due to high randomization ([7]). Methods such as multivariate microaggregation provide low usability due to the complexity introduced by its NP-hard nature ([3]). Data condensation provides an efficient solution to privacy preservation of data streams; however, the quality of data degrades as the data grows, eventually leading to low utility ([17]). Many of the multi-dimensional approaches, such as rotation perturbation and geometric perturbation, introduce high computational complexity and take unacceptably long time to execute ([18, 19]). This means that such methods in their default settings are not feasible for high dimensional data such as big data and data streams. A structured approach is needed, which can provide a practically applicable solution for selecting the best privacy preservation approach for a given application or dataset.

Several works have looked at the connection between privacy, utility, and usability. Bertino et al. proposed a framework for evaluating privacy-preserving data mining algorithms; for each algorithm, they focused on assessing the quality of the sanitized data ([17]). Other frameworks aim at providing environments for dealing with sensitive data. Sharemind is a shared multi-party computation environment allowing secret data-sharing ([20]). FRAPP is a matrix-theoretic framework aimed at helping the design of privacy-preserving random perturbation schemes ([21]). Thuraisingham et al. went one step further; they provide a vision for designing a framework that measures both the privacy and utility of multiple privacy-preserving techniques. They also provide insight into balancing privacy and utility in order to provide better privacy preservation ([22]). However, these frameworks neither provide a solution to the problem of dealing with numerous privacy preservation algorithms nor provide proper quantification of data utility and privacy for a particular application of the dataset at hand.

3. Background

Choosing the most appropriate data perturbation algorithm out of many algorithms is the primary challenge addressed by the proposed approach. The proposed framework named PPaaS aims to select the optimal perturbation algorithm that, when applied to a particular dataset, provides a proper balance between privacy and utility. This section discusses the different components involved in the conceptual development of PPaaS. Data perturbation is the process of modifying data using a certain mechanism (e.g. noise addition, geometric transformation, randomization) to prevent third parties from identifying the owners of data while performing important data analytics. A perturbed dataset's key property is its indistinguishability from the original data due to maintaining the same format. A third-party would not have the immediate impression of accessing a different dataset than the original dataset, as opposed to accessing an encrypted database. However, as all analytics are carried out on perturbed data, a certain utility level of the perturbed data must be maintained. Hence, enabling enough utility while maintaining enough privacy is the main challenge in data perturbation, in other words, proper balance between privacy and utility has to be maintained. Due to these dynamics, a perturbed dataset is expected to leak some information. A privacy model theoretically defines the level of privacy offered by a perturbation algorithm. It is important to understand the theoretical guarantees provided by a particular perturbation algorithm, as it provides an initial impression of the robustness of the corresponding perturbation algorithm. Besides, perturbed datasets can be vulnerable to data reconstruction attacks (a type of privacy attack). A data reconstruction attack tries to exploit the information leaked from a perturbed dataset with the purpose of reconstructing the original data. It is also essential to identify the resistance of a specific perturbed dataset to data reconstruction attacks, in order to provide an empirical guarantee towards the robustness of the perturbation applied to the dataset.

3.1. Perturbation Algorithms

A data perturbation algorithm's main functionality is to modify the original input data before releasing them to a third-party (e.g. an analyst) to limit possible privacy leaks or attacks by adversaries. Since the modified data are not subjected to any format conversion as in data encryption, data perturbation has lower time and space complexity compared to cryptographic approaches (e.g. fully homomorphic encryption), which are used to enforce a high level of privacy. However, for the same reason, data perturbation still results in a certain level of data leak, which needs to be carefully evaluated to restrict unanticipated privacy leaks. Data perturbation can be categorized into two classes: (1) input data perturbation, and (2) output data perturbation. Input data perturbation is also called local data perturbation, whereas output data perturbation is also called global data perturbation. As shown in Figure 2, in input data perturbation approaches (represented on the right-hand side of the figure), data perturbation is performed on the data when they leave the data owners.

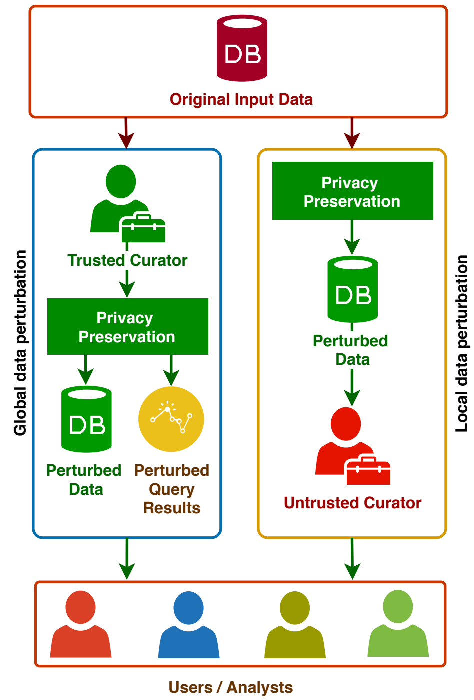

Figure 2: Global data perturbation Vs. Local data perturbation.

In output data perturbation approaches (represented on the left-hand side of the figure), a trusted curator applies data perturbation to analysis query results that were obtained by running queries on original data. Both input and output perturbation are often used. However, in an untrusted setting where only malicious parties are present, input data perturbation is preferred. Input perturbation applies higher randomization, hence input perturbation is considered to provide better privacy than output perturbation [23], while, for the same reason, output perturbation often produces better utility. Input perturbation can be divided further into unidimensional perturbation and multidimensional perturbation [6]. Additive perturbation [24], microaggregation [3], randomized response [7], rounding [25], data swapping [4] and resampling [26] are examples of unidimensional input perturbation. Data condensation [27], random rotation [18], geometric perturbation [19], random projection [28], sketch-based approach [29] are a few examples of multidimensional perturbation approaches. The merge of several forms of perturbation together is referred to as hybrid perturbation [5]. Output perturbation is achieved using approaches such as noise addition [9] and exponential mechanism [30]. In PPaaS, we use the trusted curator scenario as PPaaS wants a trusted curator to conduct the process of selecting the best perturbation instance out of multiple perturbation instances based on the dynamics of the original dataset. However, PPaaS employs input perturbation algorithms in applying perturbation over input datasets, where the perturbation is conducted over the entire dataset at once.

3.2. Privacy Models

A privacy model/privacy definition specifies the limits of private information disclosure by a certain perturbation mechanism [31]; $k-anonymity$, $l-diversity$, $(\alpha,k)-anonymity$, $t-closeness$ [32, 33] are examples of earlier privacy models. A database provides $k-anonymity$ if the data is indistinct from a minimum of $(k-1)$ records [34]. $l-diversity$ was introduced to overcome the issues of $k-anonymity$ by improving the diversity and to reduce the homogeneity of sensitive attributes [35]. A $k-anonymous$ database provides $l-diversity$ if each equivalent class has at least l well-represented values for each sensitive attribute [35]. A database provides $t-closeness$ if the difference of values between a sensitive attribute in any equivalence class and the distribution of the attributes in the whole database is no more than t [33]. However, these models and their improvements show vulnerability to privacy attacks such as minimality attack [36], composition based attacks [37], and foreground knowledge [38]. Compared to previous privacy definitions, differential privacy (DP) provides a strong privacy model that is trusted to provide a better level of privacy guarantee compared to previous privacy models [39, 40, 41, 42]. A randomized algorithm $\mathcal{A}$ satisfies ε-LDP if for all pairs of users' inputs v_1 and v_2 and for all $\mathcal{Q} \subseteq Range(\mathcal{A})$, and for $(\varepsilon \geq 0)$ Equation (1) holds. $Range(\mathcal{A})$ is the set of all possible outputs of the randomized algorithm $\mathcal{A}$.

$$\mathcal{P}r[\mathcal{A}(v_1) \in \mathcal{Q}] \leq \exp(\varepsilon)\ Pr[\mathcal{A}(v_2) \in \mathcal{Q}] \tag{1}$$

3.3. Privacy Attacks

Effective noise reconstruction techniques can significantly reduce the level of privacy in additive perturbation techniques [6]. Data perturbation approaches are vulnerable to various data reconstruction attacks such as naive estimation, independent component analysis (ICA)-based attacks, known I/O attacks, eigenanalysis, distribution analysis attacks, and spectral filtering [43, 44]. A data reconstruction attack tries to reconstruct original input data from the perturbed data. Naive estimation explores the difference between perturbed and original data. Hence, a strong perturbation can provide enough resistance to naive inference. ICA-based attacks employ independent component analysis to reconstruct the original data [43]. Known I/O attacks assume that the attacker knows/has a specific portion of the original data and knows the mapping between the known data and its corresponding perturbed data [43]. The attacker can try to use this knowledge of mapping to reconstruct the rest of the original data from the perturbed data [43]. Eigenanalysis-based attacks try to filter out the random noise from the perturbed data by analyzing the eigenvectors of the data. Spectral filtering, singular value decomposition (SVD) filtering, and principal component analysis (PCA) filtering are three examples of eigenanalysis-based attacks [16]. Distribution analysis attacks try to reconstruct the probability density function of the original data [43]. For example, microaggregation to a single variable (univariate microaggregation) is vulnerable to transparency attacks when the published data includes information about the protection method and its parameters [3]. These attacks demand that new data perturbation approaches be more robust in advanced adversarial settings to provide sufficient resilience against such attacks.

3.4. Fuzzy Inference Systems

The proposed framework (PPaaS) uses fuzzy logic to derive a final score (named as the fuzzy index: FI) to the overall quality of privacy, attack resistance, and utility of a perturbed dataset. Fuzzy logic is a logical system that provides the capability to design real-world problems as human-thinking-oriented computational paradigms. Fuzzy logic is a precise approach that allows the modeling of impreciseness of the features such as big, hot, and slow, using a multi-valued approach as opposed to classical logic that is based on 1 and 0 (binary). Hence, fuzzy logic is termed as a precise logic of imprecision. As shown in Figure 3, fuzzification, rule evaluation, and defuzzification are the three main steps of a conventional fuzzy inference system. Fuzzy logic has been used to solve many real-world problems, including fuzzy automatic transmission, hand-writing recognition, and voice recognition. Besides, fuzzy logic can be used as an effective tool for ranking-based algorithms [45, 46]. PPaaS utilizes this capability of fuzzy logic to generate ranks for the overall quality of privacy, attack resistance, and utility of a perturbed data instance.

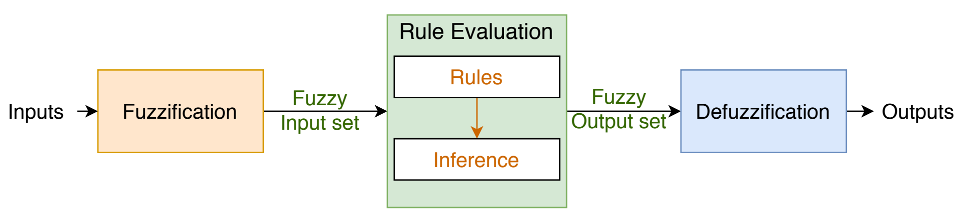

Figure 3: Basic flow of a fuzzy inference system

4. Privacy Preservation as a Service

We propose a novel approach named "Privacy Preservation as a Service (PPaaS)", a generic framework that can be used to sanitize big data in a granular and application-specific manner. In this section, we give a detailed outline of the concept. The high diversity and specificity of privacy preservation methods presents complexities, such as finding a trade-off between privacy, attack resistance, and utility. As noted in Section 2, privacy preservation algorithms can suffer from different types of biases. For example, a particular sanitization algorithm used for privacy-preserving classification may not have DM bias, but it may suffer from Type B and D biases, while another one has only Type B bias, and a third one has DM bias. Different applications may tolerate different types of bias, and there is no general rule. These biases govern the utility of a perturbed dataset. Besides, a perturbed dataset can be vulnerable to different data reconstruction attacks. It is essential to investigate the attack resistance of a perturbed dataset. A mechanism that identifies the best perturbed instance of an input dataset based on privacy, attack resistance, and utility is essential.

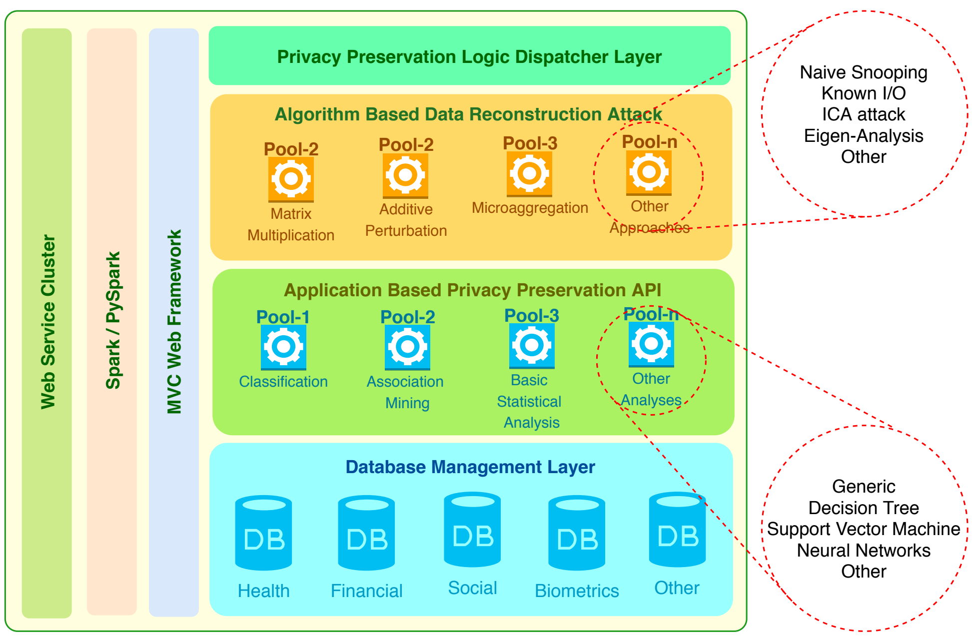

Figure 4: Privacy preservation as a service (PPaaS) for big data.

The primary intuition of PPaaS is that different privacy preservation algorithms are suitable for different data owner requirements (privacy and performance) and different data requester needs (utility and usability). A unified service of data sanitization for big data can provide an interactive solution for this problem. PPaaS can choose the most suitable privacy preservation algorithm for a particular analysis at hand. The architecture of PPaaS is presented in Figure 4. It is implemented as a web-based framework that can operate in a web service cluster. The scalability necessary for big data processing is achieved using APIs such as Spark/PySpark ([47]) (as the primary implementation language was Python) with a clean build design adapted with a Model-View-Controller (MVC) web framework. As Figure 4 shows, the framework consists of four distinct layers: (1) the database management layer, (2) the application-based privacy preservation API, (3) the algorithm-based data reconstruction attack layer, and (4) the privacy preservation logic dispatcher layer.

The privacy preservation module consists of pools of application logic (e.g. classification and association mining), and pools of privacy preservation algorithms (e.g. matrix multiplication, additive perturbation) and pools of data reconstruction attacks. The PPaaS privacy preservation module integrates a collection of privacy preservation algorithms into a collection of pools where each pool represents a particular class of data mining/analysis algorithms. The enlargement of the red circle in Figure 4 shows a possible collection of sub-pools of privacy preservation algorithms for classification. For instance, rotation perturbation (RP) ([48]) can be integrated into the "Generic" sub-pool of pool1: Classification (refer to the red circles in Figure 4), as it provides better accuracy towards a collection of classification algorithms. A particular pool may have several subdivisions to enable the synthesis of new data sanitization methods that are tailored to more specific requirements. Similarly, the pools of data reconstruction attacks are also listed to be tested against perturbed data instances of input datasets. The database management layer provides the necessary services for uniform data formatting. Figure 5 represents the pool engagements between privacy preservation algorithms (data perturbation) and data reconstruction attack approaches. During the analysis, PPaaS selects the corresponding data perturbation approaches and the related data reconstruction attacks based on the corresponding pool classifications. In the proposed concept, privacy preservation is discussed in terms of data perturbation. The following sections use "privacy preservation" and "perturbation" interchangeably, referring to the same objective.

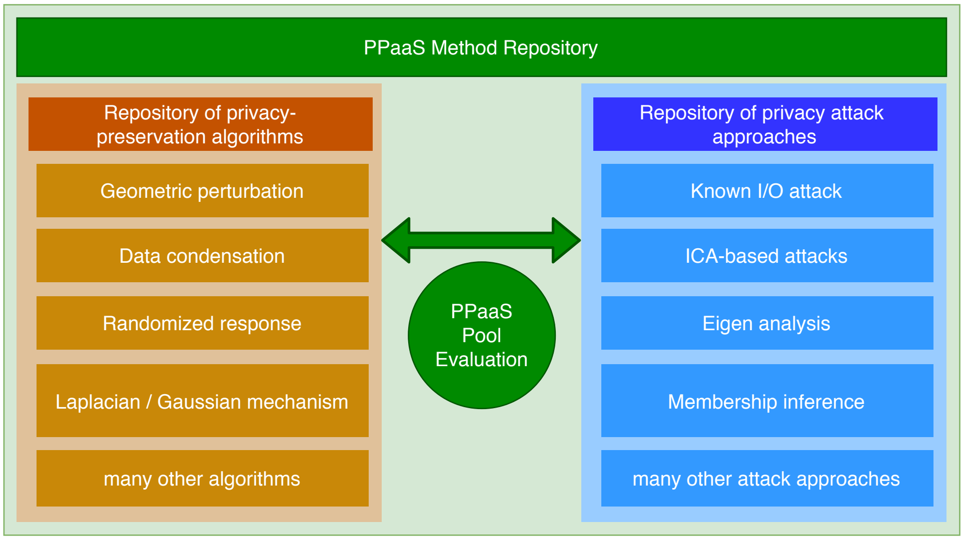

Figure 5: PPaaS Method Repository

A data owner/curator can utilize the framework to impose privacy on a particular dataset for a particular application by using the best privacy preservation approach from a pool of available algorithms. In the proposed setting, PPaaS requires a trusted curator to identify the query or the analysis requests for a given dataset, and run the PPaaS logic for the corresponding application (e.g. deep learning ([49])). The curator/data owner accesses the data and applies privacy preservation (perturbation) to the data or dataset according to the users' requirements.

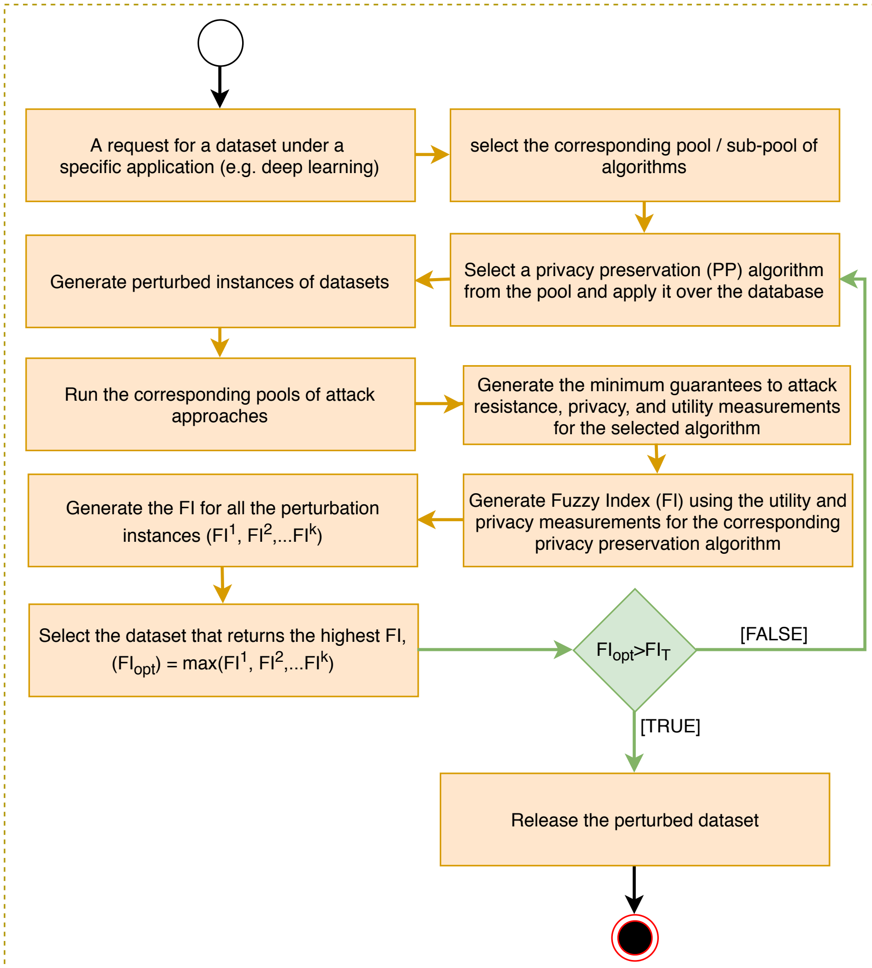

Figure 6: Flow of events in application specific privacy preservation of PPaaS

PPaaS investigates three key aspects: (1) understanding the data owner/producer requirements for privacy, (2) understanding the data requester/consumer utility needs, and (3) selecting and applying the optimum privacy-preserving algorithm to the data. Next, the perturbed instances of the input dataset are tested against the most suitable pool of data reconstruction/privacy attacks. Besed on the performance, the attack experiment analysis generates a minimum guarantee of resistance(MGR). Finally, the result of applying privacy preservation to a particular dataset is assessed using a FIS (Fuzzy Inference System) based fuzzy metric (named the fuzzy index or FI), which is a single metric to evaluate the balance between privacy and utility provided by the corresponding privacy preservation algorithm. Fig 6 shows the main flow of PPaaS in releasing a perturbed dataset with a customized application of privacy-preservation. The data curator will receive a request for a certain operation on the underlying dataset. For example, this request can be for deep learning on a medical dataset that is maintained by the corresponding data owner. The data owner forwards the request to the PPaaS framework, which will select the corresponding pool/sub-pool of privacy preservation algorithms allocated under deep learning. In the example, this pool may include the following algorithms: local differentially private approaches, geometric data perturbation approaches, random projection-based data perturbation approaches, which are suitable for producing high utility for deep learning. Next, PPaaS sequentially applies the corresponding pool of privacy preservation algorithms. Then, PPaaS runs the corresponding pool of data reconstruction algorithms on each of the perturbed data instances to generate a minimum guarantee of attack resistance for each perturbed dataset. Based on the results, PPaaS generates a fuzzy index for each perturbed data instance (perturbation algorithm). If a particular pool has four privacy preservation algorithms, PPaaS will produce for perturbed data instances, which will result in 4 FI values. Next, the PPaaS will select the perturbed dataset with the highest FI, because the corresponding dataset provides the best balance between privacy, attack resistance, and utility.

PPaaS uses a fuzzy inference system (FIS) to generate the fuzzy index. Privacy, minimum guarantee of attack resistance (MGR) and utility are the only inputs to the FIS that generates a final score, that is, the fuzzy index (FI). FI is a quantitative rank that rates the complete process of privacy preservation upon a particular dataset for a given application. A heuristic approach was followed in defining the fuzzy rules that focused on maintaining a balance between privacy, attack resistance, and utility. The universe of discourse of the inputs and output ranges from 0 to 1. A higher FI value suggests that the final dataset has high privacy, attack resistance, and utility with a good balance between them. The PPaaS dispatcher investigates the value of FI corresponding to a particular process of sanitization, compares it with a user-defined balance guarantee, FI_T that is taken as an input parameter from the data owner. If $FI_{opt} \geqslant FI_T$, the dataset will be released to the data requester, where FI_{opt} is the maximum FI generated by the pool. Otherwise, the PPaaS will reapply the random perturbation algorithm to find a better solution that satisfies FI_T requirement.

A fuzzy inference system (FIS) takes several inputs and generates a certain output based on evaluating a collection of specified rules, which are expressed as fuzzy rules (refer to Section 3.4). In an FIS, the first step is to apply fuzzification to the input variables. Fuzzification maps inputs to values from 0 to 1 using a collection of membership functions. There are different types of membership functions that can be used for this step. Triangular, Trapezoidal, Piecewise linear, Gaussian, and Singleton are some examples of such membership functions. The most suitable membership function and its range and shape for a particular problem need to be selected based on the problem's dynamics in the corresponding environment and the domain expert's knowledge. Figure 7 represents the fuzzification of an input variable (e.g. privacy) using two membership functions that represent two levels (LOW and HIGH) of the input. LOW is represented using a triangular membership function, whereas HIGH is represented using a Trapezoidal MF. In this plot (refer to Figure 7), μ_{input} quantifies the corresponding input's (x_i) degree of membership.

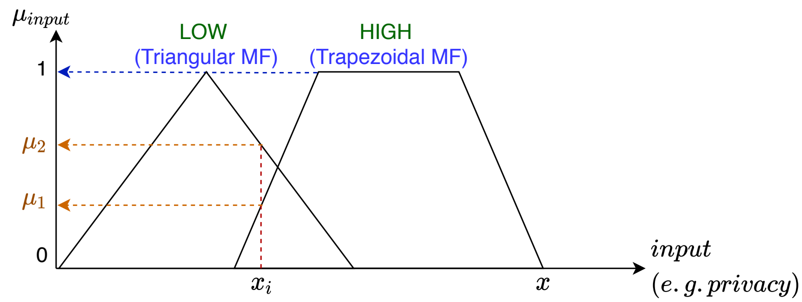

Figure 7: An instance of an input variable definition

We need to first map privacy, attack resistance, and utility into fuzzy memberships. Figure 7 shows an example of mapping input values into fuzzy memberships, which allows the activation of rules that are in terms of linguistic variables. During this process, the membership functions allow the fuzzifier to determine the degree to which the input values belong to each membership function. For example, the figure shows the fuzzification of the input value, x_i, which produces the two fuzzified membership (degree of membership) values , LOW(x_i)=μ_1 and HIGH(x_i)=μ_2. Figure 11 shows the mapping of the three inputs: privacy, attack resistance, utility, and the output: FI into fuzzy memberships (A more detailed explanation on PPaaS input and output fuzzification is included later).

As shown in Figure 3, the second module of an FIS is the rule evaluation, which involves inference based on a collection of linguistic rules, which is also called the rule base. A rule is defined using the IF-THEN convention (e.g. IF $input_1$ is HIGH AND $input_2$ is LOW THEN output is MEDIUM). As shown in Figure 11, we can identify that all inputs and outputs of PPaaS has three levels of memberships (LOW, MEDIUM, HIGH). Assume that a particular FIS has two inputs ($input_1$ and $input_2$) and one output. If $input_1$ and $input_2$ have two membership levels each (e.g. LOW and HIGH) and the output with the levels LOW, MEDIUM, and HIGH, we can define an example rule-base, which is shown in Figure 8. Each box in the figure represents a rule where the value of a box represents the output variable's membership for the corresponding rule.

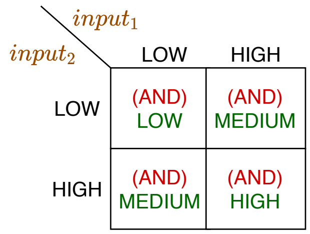

Figure 8: Example rule base of a fuzzy inference system

The rule evaluation step (the inference engine) combines all fuzzy conclusions obtained by inferencing the rules into a single conclusion. Each inference will suggest a different action. A simple MAX-MIN (OR-AND) operation of the selection can be used where the maximum fuzzy value of the inference is generally used as the final conclusion. For simplicity, let us consider both $input_1$ and $input_2$ have the same function definitions represented in Figure 7. Let us consider that δ_1, δ_2 are the two membership values returned by LOW and HIGH membership functions of $input_2$ and $\delta_1 < \mu_1 < \mu_2 < \delta_2$. Assuming that we consider the AND (equivalent to MIN) operation between antecedents, we can obtain the final values for the output membership functions as represented in Figure 9. For the rules with the same consequent, the OR (MAX) operation between the corresponding consequent values is generally considered.

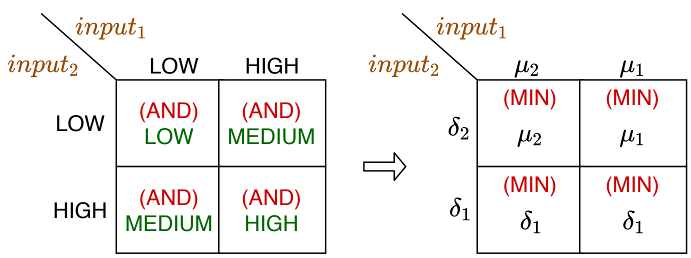

Figure 9: Rule evaluation based on the rule base

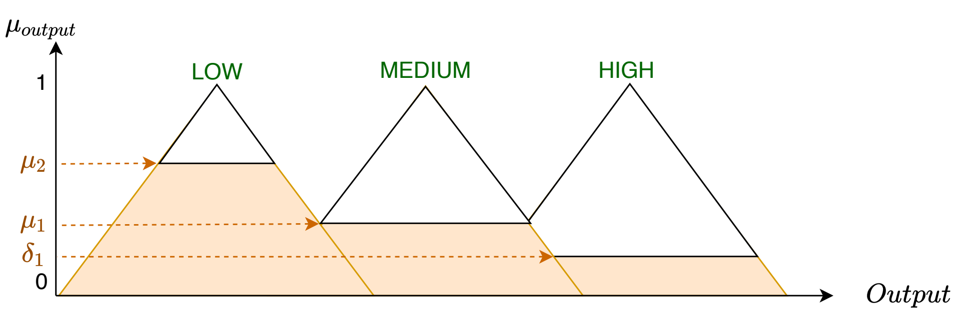

Figure 10: Rule aggregation on the output variable

As shown in Figure 10, we can use clipping (α-cut) and aggregate the rules to obtain the colored area in the membership levels in the output variable. In this figure, μ_{output} represents the output membership values (μ_1, μ_2, and δ_1) obtained by the rule evaluation process (as depicted by Figure 9). The final step of the fuzzy inference system is to apply defuzzification based on the aggregated shape of the output function. There are several defuzzification approaches; however, the most popular approach is the centroid-based technique, which finds the point where a vertical line would slice the aggregate set into two equal masses (the center of gravity: COG). Equation 2 can be used to obtain this value (COG), where $x = output$ and $\mu_x = \mu_{output}$ (refer to Figure 10).

$$COG = \frac{\int_{min}^{max} \mu_x x dx}{\int_{min}^{max} \mu_x dx} \tag{2}$$

In the proposed framework (PPaaS), we define a FIS to take the three inputs: the minimum guarantee of privacy, MGR, and the minimum guarantee of utility to produce an output named fuzzy index (FI). FI provides an impression of the quality of the balance between privacy, attack resistance, and utility generated after perturbing a dataset using a privacy preservation algorithm. According to the domain knowledge, we already know that a good privacy preservation algorithm should enforce high privacy, high attack resistance while producing good utility (e.g. accuracy). Following this notion, FI should ideally provide high values only when the minimum guarantee of privacy, MGR, and the minimum guarantee of utility are high. In case one is high and the other is low, the FI should be a lower value. Hence, the fuzzy model should produce a rule-surface, as presented in Figure 12. Considering all these dynamics between privacy, utility, and FI, we introduced three membership functions (LOW, MEDIUM, HIGH) for each variable. Next, we considered Gaussian functions for all the membership functions in the two input variables and output variables, as shown in Figure 11. Finally, we defined the eleven rules given in Equation 3 to obtain the rule-surface depicted in Figure 12. As defined in the first three rules, the fuzzy inference engine will generate a low value for FI when any one of the three input parameters ("privacy", "attack_resistance", and "utility") take a low input value. This is to ensure that the lower the value of any one of the input parameters, the lower the FI value. Consequently, the remaining rules do not consider any rule combination where any one of the input parameters is LOW. Rule 5 to Rule 11 consider the dynamics of FI under all remaining combinations of membership levels ($MEDIUM$ and $HIGH$). For example, Rule 4 considers the situation where "privacy", "attack_resistance", and "utility" are $MEDIUM$, $MEDIUM$, and $MEDIUM$ respectively, and under this situation FI is considered to result in $MEDIUM$ values.

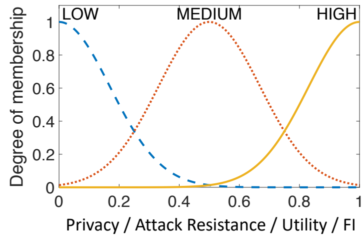

Figure 11: Fuzzy membership functions of the input/output variables

$$
\begin{array}{l}
\textbf{Rule 1: } IF(privacy = LOW)\ THEN\ (FI = LOW) \\
\textbf{Rule 2: } IF(attack\_resistance = LOW)\ THEN\ (FI = LOW) \\
\textbf{Rule 3: } IF(utility = LOW)\ THEN\ (FI = LOW) \\
\textbf{Rule 4: } IF(privacy = MEDIUM\ AND\ attack\_resistance = MEDIUM\ AND\ utility = MEDIUM)\ THEN\ (FI = MEDIUM) \\
\textbf{Rule 5: } IF(privacy = MEDIUM\ AND\ attack\_resistance = MEDIUM\ AND\ utility = HIGH)\ THEN\ (FI = MEDIUM) \\
\textbf{Rule 6: } IF(privacy = MEDIUM\ AND\ attack\_resistance = HIGH\ AND\ utility = MEDIUM)\ THEN\ (FI = MEDIUM) \\
\textbf{Rule 7: } IF(privacy = MEDIUM\ AND\ attack\_resistance = HIGH\ AND\ utility = HIGH)\ THEN\ (FI = HIGH) \\
\textbf{Rule 8: } IF(privacy = HIGH\ AND\ attack\_resistance = MEDIUM\ AND\ utility = MEDIUM)\ THEN\ (FI = MEDIUM) \\
\textbf{Rule 9: } IF(privacy = HIGH\ AND\ attack\_resistance = MEDIUM\ AND\ utility = HIGH)\ THEN\ (FI = HIGH) \\
\textbf{Rule 10: } IF(privacy = HIGH\ AND\ attack\_resistance = HIGH\ AND\ utility = MEDIUM)\ THEN\ (FI = HIGH) \\
\textbf{Rule 11: } IF(privacy = HIGH\ AND\ attack\_resistance = HIGH\ AND\ utility = HIGH)\ THEN\ (FI = HIGH)
\end{array}
\tag{3}
$$

Figure 12 depicts the rule surface of the fuzzy inference system (FIS), which is used to generate FI. It shows the change of FI when any two of the inputs ("privacy", "attack_resistance", and "utility") are varied while the third input is kept constant. Consequently, when Input 1 is "privacy", Input 2 can be "attack_resistance" or "utility". As shown in the figure, FIS generates higher values for FI when both utility and privacy are high, whereas for lower values of privacy and utility FI also stays at a lower level. As shown in the figure, the rule surface makes sure that a higher value of only one parameter (privacy, MGR, or utility) does not result in a higher value for FI. This property guarantees that the proposed PPaaS framework maintains a good balance between privacy, MGR, and utility.

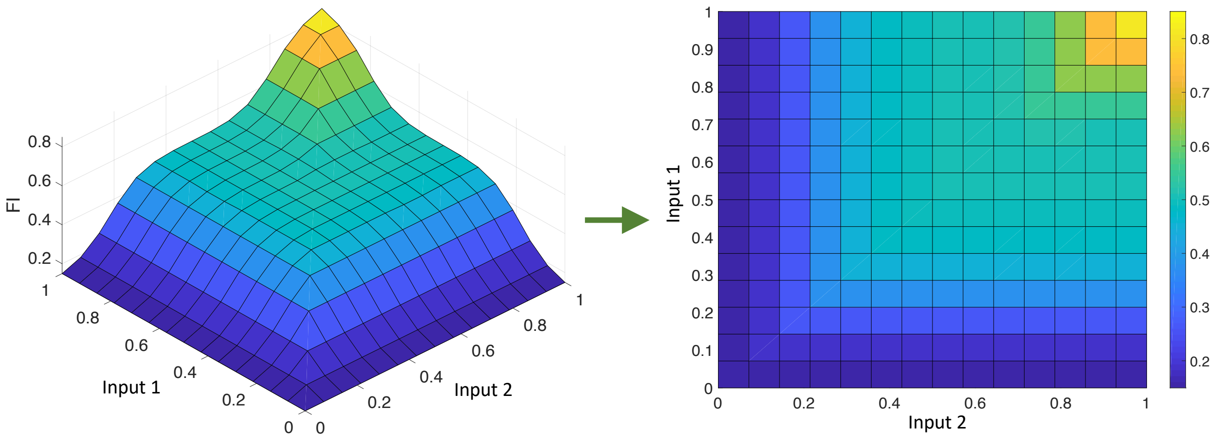

Figure 12: Rule surface of the FIS

4.1. Privacy Metric (Privacy Quantification)

During the application of each privacy preservation algorithm, the privacy will be quantified empirically using a multi-column privacy metric, considering that the input datasets are n-dimensional matrices. In the proposed setting, we assume that all the attributes of a particular dataset are equally important, and we ensure it by applying z-score normalization to the input datasets. Then we calculate the differential entropy between the perturbed and non-perturbed attributes of the datasets. The correlation of the data distributions of original data and reconstructed data can be effectively used to extract private information by guessing original data with a higher level of accuracy. Consequently, it is essential to take the data's underlying distribution into account when quantifying the inherent privacy [50]. Differential entropy of a random variable provides an effective mechanism to quantify privacy by considering such side-information into account. The differential entropy $h(A)$ of a random variable A is defined as given in Equation 5. $h(A)$ can effectively be used to measure the privacy of a random variable [16]. Ω_A denotes the domain of A. $h(A)$ measures the uncertainty inherent in the value of A. $2^{h(A)}$ is proposed to measure the privacy inherent in the random variable A. This value (refer to Equation 4) is also denoted by $\prod(A)$, where $f_A(a)$ is the density function of A.

$$\prod(A) = 2^{h(A)} \tag{4}$$

$$h(A) = -\int_{\Omega_A} f_A(a) log_2 f_A(a) da \tag{5}$$

Given a random variable B, the conditional differential entropy of A is defined according to Equation

6.

$$h(A|B) = -\int_{\Omega_{A,B}} f_{A,B}(a,b) log_2 f_{A|B=b}(a) da db \tag{6}$$

Therefore, $\prod(A|B) = 2^{h(A|B)}$ denotes the average conditional privacy of A given B. We can use $\prod(A|B)$ to investigate the privacy of an attribute (a data series) after the perturbed version of that attribute is released to a third party. Therefore, the conditional privacy loss of A, given B, $P(A|B)$ can be given according to Equation 8. $I(A;B)$ is known as the mutual information between the random variables A and B, where $I(A;B)$ is given in Equation 7.

$$I(A;B) = h(A) - h(A|B) = h(B) - h(B|A) \tag{7}$$

Let's consider A to be the original variable and B to be the perturbed version A. $P(A|B)$ provides the fraction of privacy of A which is lost by revealing B.

$$P(A|B) = 1 - \prod(A|B)/\prod(A) = 1 - 2^{h(A|B)}/2^{h(A)} = 1 - 2^{-I(A;B)} \tag{8}$$

Assume that, $B = A+N$, where N represents the noise variable, which is independent of A. Since, A and N are independent, $h(B|A) = h(N)$. Consequently, we can represent $P(A|B)$ using Equation 9 as $I(A;B) = h(B) - h(N)$.

$$P(A|B) = 1 - 2^{-(h(B)-h(N))} \tag{9}$$

From Equation 8, $\prod(A|B)$ (the privacy of A after revealing B) can be obtained using Equation 10.

$$\prod(A|B) = \prod(A) \times (1 - P(A|B)) \tag{10}$$

The lower the value of $\prod(A|B)$, the lower the privacy of A, when B is released. Hence, for a given dataset we consider the minimum of $\prod(X_i|X_i^p)$ returned by all the attributes to identify the minimum privacy guarantee (where X_i^p represents the perturbed version of the attribute, X_i). To obtain $\prod(X_i|X_i^p)$, we should know the density function of X_i. For this purpose, we used an approach which declares a certain number of bins within the range of 0 to 1. Next, we assign the values of a particular variable to each bin and find the probability of each bin using the number of values assigned, as shown in Algorithm 1, which is used to generate the inherent uncertainty of a particular attribute

($h(attribute)$).

Algorithm 1: Generating the inherent uncertainty $h(X)$ of an attribute ($X(attribute)$)

Input:
$X \leftarrow$ attribute
$bw \leftarrow$ bin window size (default: 0.01)

Output: $h(X) \leftarrow$ the inherent uncertainty of X

declare bn (bins) from 0 to 1 with an window interval of bw;
normalize X between 0 to 1;
assign the values of X to the bins in bn;
count the number of values assigned to each bin in bn;
generate the density function of X ($f_X(x)$) by calculating the bin probabilities of each bin in bn;
$h(A) = -\int_{\Omega_A} f_A(a) log_2 f_A(a) da$;
return $h(A)$

We use Algorithm 2 to generate the minimum privacy guarantee of a perturbed dataset.

Algorithm 2: Generating minimum empirical privacy guarantee $min \prod(X_i|X_i^p)$ for a perturbed dataset

Input:
$D \leftarrow$ original dataset with n number of attributes
$D^p \leftarrow$ a perturbed instance of the original dataset, D
$bw \leftarrow$ bin window size

Output: $min\{\prod(X_i|X_i^p)\}_{i=1}^n \leftarrow$ the minimum empirical privacy guarantee

for *each attribute* X_i *and its perturbed attribute,* X_i^p **do**
    $noise_{X_i} = X_i^p - X_i$;
    $h(X_i) = Algorithm1(X_i)$;
    $h(X_i^p) = Algorithm1(X_i^p)$;
    $h(noise_{X_i}) = Algorithm1(noise_{X_i})$;
    $I(X_i; X_i^p) = h(X_i^p) - h(noise_{X_i})$;
    $P(X_i|X_i^p) = 1 - 2^{(-I(X_i;X_i^p))}$;
    $\prod(X_i) = 2^{h(X_i)}$;
    $\prod(X_i|X_i^p) = \prod(X_i) \times (1 - P(X_i|X_i^p))$;
return $min\{\prod(X_i|X_i^p)\}_{i=1}^n$

4.2. Attack Resistance Quantification

During the attack resistance quantification, PPaaS runs the corresponding pool of data reconstruction attacks on the perturbed instances. For example, if the pool of perturbation algorithms contain j number of perturbation algorithms, and the pool of data reconstruction attacks contain k number of approaches, testing all k attacks on j perturbation instances of input dataset will produce $j \times k$ number of reconstructed data instances for a given dataset. In the proposed setting, we assume that all the attributes of a particular dataset are equally important, and we make it sure by applying z-score normalization to the input datasets. After generating each reconstructed data instances, we measure the variance, $V(P)$ (where $P = (X^r - X)$) between the attributes of the corresponding reconstructed data instance and the original dataset. The more different the reconstructed data from original data, the better the perturbation has been. $Var(P)$ provides an effective mechanism to capture this notion [18]. Hence, the higher the $Var(P)$, the higher the difficulty in reconstructing original data from perturbed data. If X^r is a reconstructed data series of attribute X, the level of strength of the perturbation method can be measured using $Var(P)$, where $P = (X^r - X)$. $Var(P)$ can be given by Equation 11.

$$Var(P) = Var(p_1, p_2, \ldots, p_n) = \frac{1}{n}\sum_{i=1}^{n}(p_i - \bar{p})^2 \tag{11}$$

Next, the attribute having the minimum of all $Var(P)$ (hence the minimum difference between the corresponding attribute) is considered as the most vulnerable attribute of the dataset (or the most successfully reconstructed attribute). The higher the $Var(P)$, the higher the strength of the corresponding attribute, as $Var(P)$ indicates the difficulty of estimating the original data from the perturbed data ([2]). Equation 12 shows the generation of the minimum variance, $Var(P)^{min}$) for a particular reconstructed dataset instance.

$$Var(P)^{min} = min\{Var(P_1), Var(P_2), \ldots Var(P_n)\} \tag{12}$$

In this way, PPaaS will produce t number of $Var(P)_{min}$ values if t number of data reconstruction attacks are being tested on a single perturbed instance of the input dataset. From these t instances, we select the minimum variances $Var(P)_{min}$ value, which represents the minimum guarantee of attack resistance of a particular perturbed data instance, as shown in Equation 13.

$$Var(P)_{min} = min\{Var(P)_1^{min}, Var(P)_2^{min}, \ldots Var(P)_t^{min}\} \tag{13}$$

Finally, we scale the $Var(P)_{min}$ values within 0 and 1, by applying Equation 14 to the corresponding pool. The value returned from Equation 14 is considered as the input to the FIS (which accepts inputs of range: $[0, 1]$).

$$resistance\_input = \frac{Var(P)^{i}_{min}}{max\{Var(P)^{1}_{min}, \ldots, Var(P)^{n}_{min}\}} \tag{14}$$

4.3. Utility Quantification

The accuracy of the results produced by the requested service is evaluated experimentally to generate the empirical utility. If the application being examined is classification, the classification accuracy is generated for all the privacy preservation algorithms in the pool for the corresponding type of data classification. However, if the corresponding pool of applications contains more than one application to be tested, the minimum accuracy (the minimum guarantee of utility) returned by the corresponding perturbed data instance is considered.

For the experimental evaluation of PPaaS we consider only data classification. The utility of data classification can be quantified based on different metrics such as precision, recall, F-measure, accuracy [51]. Any one of these metrics should provide a reasonable approach to measure the utility of a data classification result. It is the application that determines which one of these is the most suitable metric. PPaaS chooses the best perturbed dataset by considering all privacy preservation approaches' relative performance on an input dataset. Hence, the primary requirement of PPaaS is to use only one suitable metric for the utility quantification. For the experimental analysis of PPaaS, we chose classification accuracy measured using Equation 15 (where TP = the number of true positives, TN = the number of true negatives, FP = the number of false positives, FN = the number of false negatives) for the utility quantification of the privacy preservation approaches.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + FN + TN)} \tag{15}$$

4.4. Algorithm for generating FI

Algorithm 3 is used for generating FI for a particular pool of privacy preservation algorithms.

Algorithm 3: Algorithm for generating FI for a pool of algorithms

Input:
$D \leftarrow$ input dataset
$[pp_1, pp_2, \ldots, pp_n] \leftarrow$ pool of privacy algorithms

Output:
$BD_i \leftarrow$ selected perturbed dataset
$pp_i \leftarrow$ selected privacy preserving algorithm

perturb D using the pool of algorithms to generate $D_1^p, D_2^p, \ldots, D_n^p$;
for *each perturbed dataset,* D_i^p **do**
    generate minimum privacy guarantee (pi_i) using Algorithm 2;
    generate minimum attack resistance guarantee (vp_i) using Equation 14;
    generate minimum utility guarantee (u_i) by running the corresponding application pool on D_i^p (refer to Section 4.3);
    generate the fuzzy index (FI_i) by considering pi_i, vp_i and u_i, as inputs to the fuzzy inference system (FIS);
select the dataset (BD_i) that returns the highest FI;

5. Results

In this section, we provide the results of PPaaS in selecting the best perturbed dataset from a particular pool of algorithms. During the experiments, we consider five classification algorithms: Multilayer perceptron (MLP), k-nearest neighbor (IBK), Sequential Minimal Optimization (SVM), Naive Bayes, and J48 ([52]). We use four privacy preservation algorithms: rotation perturbation (RP), geometric perturbation (GP), PABIDOT, and SEAL ([2]), which are benchmarked for utility for the selected classification algorithms ([2]). The algorithms were tested on five different datasets retrieved from the UCI machine learning data repository[1]. Table 1 provides a summary of the datasets. For the generation of the minimum guarantee to attack resistance, we used three data reconstruction attacks: (1) naive estimation (naive inference), (2) Known I/O attack [6], and (3) ICA (Independent Component Analysis)-based attacks [6]. The corresponding data reconstruction attacks were run on the perturbed instances, and the standard deviation of the difference between the normalized original data and the reconstructed data for each instance was recorded. For the known I/O attack, we assumed that 10% of the original data is known to the adversary. We set the number of iterations to 10 for both RP and GP with a noise factor (sigma) of 0.3 (the default setting). During the experiments, we used

[1]http://archive.ics.uci.edu/ml/index.php

a noise standard deviation (σ) of 0.3 for PABIDOT, whereas an ϵ of 1 was maintained for SEAL. All the experiments were run on a Windows 7 (Enterprise 64-bit, Build 7601) computer with an Intel(R) i7-4790 (4^{th} generation) CPU (8 cores, 3.60 GHz) and 8GB RAM.

Table 1: A summary of the datasets used for the experiments.

Dataset	Abbreviation	Number of Records	Number of Attributes	Number of Classes
Wholesale customers[2]	WCDS	440	8	2
Wine Quality[3]	WQDS	4898	12	7
Page Blocks Classification [4]	PBDS	5473	11	5
Letter Recognition[5]	LRDS	20000	17	26
Statlog (Shuttle)[6]	SSDS	58000	9	7
HEPMASS[7]	HPDS	3310816	28	2
HIGGS[8]	HIDS	11000000	28	2

In the proposed experimental setting, we consider 25 case studies where each case study considers one of the five classification algorithms and one of the five datasets. We consider a pool of four data perturbation algorithms: RP, GP, PABIDOT, and SEAL; (CS stands for "case study") as shown in Tables 2 and 4. Next, we evaluated the performance of each privacy preservation algorithm in each case to generate the ranks (Fuzzy Indices: FIs) and recorded them in Table 4.

Table 2: Classification accuracies returned by four privacy-preserving algorithms and five different classification algorithms, and the minimum privacy guarantees generated according to Equations 12 and 14 using the differences between original and perturbed data. (CS: case study)

Dataset	Privacy preserving algorithm	Utility after privacy preservation					Privacy guarantee	
		MLP CS 1	IBK CS 2	SVM CS 3	Naive Bayes CS 4	J48 CS 5	$min(\prod(X_i\|X_i^p))_{i=1}^n$	Scaled $min(\prod(X_i\|X_i^p))_{i=1}^n$
LRDS	RP	0.7404	0.8719	0.7107	0.4841	0.6489	1.0160	0.9981
	GP	0.7912	0.9305	0.7792	0.5989	0.7054	1.0169	0.9990
	PABIDOT	0.7822	0.9224	0.7848	0.6280	0.7262	1.0179	1.0000
	SEAL	0.8059	0.9367	0.8171	0.6310	0.8528	1.0157	0.9978
PBDS	RP	0.9200	0.9552	0.8999	0.3576	0.9561	0.9988	0.9979
	GP	0.9024	0.9567	0.8993	0.4310	0.9549	1.0009	1.0000
	PABIDOT	0.9583	0.9476	0.9209	0.8968	0.9492	0.9927	0.9838
	SEAL	0.9634	0.9673	0.9559	0.8697	0.9634	0.9974	0.9965
SSDS	RP	0.9626	0.9980	0.8821	0.6904	0.9951	0.9991	0.9992
	GP	0.9873	0.9981	0.7841	0.7918	0.9959	0.9999	1.0000
	PABIDOT	0.9865	0.9867	0.9280	0.9134	0.9874	0.9920	0.9921
	SEAL	0.9970	0.9921	0.9851	0.8994	0.9987	0.9961	0.9962
WCDS	RP	0.8909	0.8500	0.8227	0.8455	0.8682	1.0078	0.9974
	GP	0.9182	0.8659	0.8500	0.8432	0.8886	1.0078	0.9974
	PABIDOT	0.9045	0.8545	0.8841	0.8886	0.8841	1.0104	1.0000
	SEAL	0.8932	0.8682	0.8909	0.8841	0.8659	1.0072	0.9968
WQDS	RP	0.4765	0.5329	0.4488	0.3232	0.4553	1.0268	1.0000
	GP	0.4886	0.5688	0.4488	0.3216	0.4643	1.0267	0.9999
	PABIDOT	0.5412	0.6182	0.5147	0.4657	0.4916	1.0225	0.9958
	SEAL	0.5392	0.6402	0.5202	0.4783	0.8415	1.0255	0.9958

Table 2 shows the classification accuracy and the minimum privacy guarantee produced for each

[2]https://archive.ics.uci.edu/ml/datasets/Wholesale+customers
[3]https://archive.ics.uci.edu/ml/datasets/Wine+Quality
[4]https://archive.ics.uci.edu/ml/datasets/Page+Blocks+Classification
[5]https://archive.ics.uci.edu/ml/datasets/Letter+Recognition
[6]https://archive.ics.uci.edu/ml/datasets/Statlog+%28Shuttle%29
[7]https://archive.ics.uci.edu/ml/datasets/HEPMASS#
[8]https://archive.ics.uci.edu/ml/datasets/HIGGS#

pool of privacy preservation algorithms. During the minimum privacy guarantee generation, we used a bin size of 0.01 (the default value) in Algorithm 1. In each pool, the input datasets were perturbed using the four privacy preservation algorithms. Then the perturbed data were analysed by each classification algorithm to generate classification accuracy (utility) values. Table 3, includes the $\sqrt{min(Var(P))}$ values generated during the attack resistance analysis.

Table 3: Analysis on the minimum attack resistance guarantee.

Dataset	**Privacy-preserving algorithm**	$\sqrt{Var(P)_{min}}$ **values returned under each attack**				
		NI	**ICA**	**I/O**	$\sqrt{Var(P)_{min}}$	**scaled** $\sqrt{Var(P)_{min}}$
LRDS	RP	0.8750	0.4057	0.0945	0.0945	0.1353
	GP	1.3248	0.6402	0.0584	0.0584	0.0836
	PABIDOT	1.4046	0.7038	0.6982	0.6982	0.9994
	SEAL	1.4061	0.7024	0.6986	0.6986	1.0000
PBDS	RP	0.7261	0.5560	0.0001	0.0001	1.4426e-04
	GP	0.2845	0.1525	0.0000	0.0000	0.0000
	PABIDOT	1.4102	0.6951	0.6755	0.6755	0.9745
	SEAL	1.3900	0.7008	0.6932	0.6932	1.0000
SSDS	RP	1.2820	0.1751	0.0021	0.0021	0.0030
	GP	1.4490	0.0062	0.0011	0.0011	0.0016
	PABIDOT	1.4058	0.7069	0.7031	0.7031	1.0000
	SEAL	1.4065	0.7038	0.7027	0.7027	0.9994
WCDS	RP	1.0105	0.6315	0.0000	0.0000	0.0000
	GP	1.4620	0.1069	0.0000	0.0000	0.0000
	PABIDOT	1.3680	0.6771	0.6512	0.6512	0.9931
	SEAL	1.3130	0.6775	0.6557	0.6557	1.0000
WQDS	RP	1.2014	0.4880	0.0057	0.0057	0.0083
	GP	1.3463	0.3630	0.0039	0.0039	0.0057
	PABIDOT	1.4019	0.7034	0.6901	0.6901	1.0000
	SEAL	1.3834	0.7018	0.6859	0.6859	0.9939

The values in Tables 2 and Table 3 are evaluated using the proposed fuzzy model to generate the ranks for each privacy preservation algorithm and perturbed dataset as given in Table 4. The highest ranks generated in each pool of algorithms are in bold and highlighted in colour. Although SEAL has the best performance results in many cases, the table clearly shows that the input dataset and the choice of application (e.g. classification) are also important when selecting the best privacy preservation approach. Consequently, this result does not mean that SEAL will outsmart other algorithms in other applications with other datasets.

Table 4: The best choice of perturbation in each pool based on the highest FI rank values returned.

Dataset	Privacy-preserving algorithm	FI rank values returned in each Case Study (CS)				
		MLP CS 1	IBK CS 2	SVM CS 3	Naive Bayes CS4	J48 CS 5
LRDS	RP	0.2744	0.2722	0.2744	0.2523	0.2744
	GP	0.2023	0.2005	0.2023	0.2023	0.2023
	PABIDOT	0.8104	0.8471	0.8115	**0.8390**	0.8068
	SEAL	**0.8190**	**0.8479**	**0.8230**	0.8378	**0.8338**
PBDS	RP	0.1496	0.1496	0.1496	0.1496	0.1496
	GP	0.1495	0.1495	0.1495	0.1495	0.1495
	PABIDOT	0.8407	0.8402	0.8378	0.8343	0.8403
	SEAL	**0.8490**	**0.8491**	**0.8487**	**0.8375**	**0.8490**
SSDS	RP	0.1505	0.1505	0.1505	0.1505	0.1505
	GP	0.1500	0.1500	0.1500	0.1500	0.1500
	PABIDOT	0.8479	0.8479	0.8453	**0.8437**	0.8479
	SEAL	**0.8495**	**0.8493**	**0.8492**	0.8431	**0.8501**
WCDS	RP	0.1495	0.1495	0.1495	0.1495	0.1495
	GP	0.1495	0.1495	0.1495	0.1495	0.1495
	PABIDOT	**0.8428**	0.8325	0.8393	0.8402	**0.8393**
	SEAL	0.8422	**0.8372**	**0.8418**	**0.8406**	0.8367
WQDS	RP	0.1522	0.1523	0.1522	0.1522	0.1522
	GP	0.1513	0.1514	0.1513	0.1513	0.1513
	PABIDOT	**0.8486**	**0.8398**	**0.8491**	0.8368	**0.8477**
	SEAL	0.8480	0.8342	0.8485	**0.8426**	0.8292

5.1. Scalability of PPaaS

It is important that PPaaS runs in an high performance computing-based environment, as it involves multiple processing modules and heavy computation. We can identify two modules of PPaaS: (1) Generating perturbed instances and (2) Running the data reconstruction attacks as those needing the most computation. As shown in Figure 6, PPaaS can run the steps multiple times until the FI value reaches a certain threshold (FI_T). The parallel processing capabilities of implementational components, such as PySpark, allow PPaaS to utilize independent processing modules efficiently. It is also essential that the privacy-preservation approaches used in the PPaaS method repository are efficient enough and capable of dealing with high-dimensional data (e.g. big data). Table 5 shows the performance (scalability) of the privacy-preservation algorithms when they are applied to high-dimensional data. For this experiment, we used an SGI UV3000 supercomputer, with 64 Intel Haswell 10-core processors, 25MB cache, and 8TB of global shared memory connected by SGI's NUMAlink interconnect. As shown in Table 5, PBDOT and SEAL perform extremely well compared to RP and GP, and may be the preferred privacy-preservation algorithms under complex scenarios such as the privacy preservation of big data.

Table 5: Efficiency of the privacy preservation algorithm used in the experiments when they are introduced to high-dimensional data

Dataset	Dimensions	RP	GP	PABIDOT	SEAL (ws = 10,000)
HPDS	3310816×28	Not converged for 100 hours	Not converged for 100 hours	2.9 hours	97.82 seconds
HIDS	11000000×28	Not converged for 100 hours	Not converged for 100 hours	11.16 hours	1.02E+03 seconds

6. Discussion

In this paper, we proposed a new paradigm named privacy preservation as a service (PPaaS), to improve the process of privacy preservation of a dataset or application, eventually improving the utility of existing and new privacy preservation approaches. The domain of data privacy contains a plethora of different privacy preservation approaches that have been proposed for different types of applications. However, there are still challenges when it comes to identifying the best privacy preservation method for a given dataset and a certain application; in particular, providing the best utility and maintaining privacy at a high level is difficult. Consequently, it is a highly complex process to identify the best possible privacy preservation approach for a particular application. PPaaS provides a solution by introducing a service-oriented framework that collects existing privacy preservation approaches and semantically categorizes them into pools of applications. Developers of new privacy preservation algorithms can introduce their methods to the PPaaS framework and add to the corresponding pools of applications. When a data owner/curator wants to apply privacy-preservation to a particular dataset, PPaaS will rank the methods in the relevant pools of applications with respect to the dataset. The ranks are expressed in the form of a Fuzzy Index (FI). FI values are generated using a fuzzy inference system that takes three inputs: privacy (the minimum privacy guarantee), attack resistance (the minimum guaranteed attack resistance), and utility (the minimum guaranteed utility). PPaaS quantifies privacy using a metric ($\prod(X_i|X_i^p)$) based on differential entropy of input data and perturbed data. PPaaS considers the concept of minimum privacy guarantee ($min\{\prod(X_i|X_i^p)\}_{i=1}^n$), where the minimum of $\prod(X_1|X_1^p)$ to $\prod(X_n|X_n^p)$ is considered. The strength of the weakest attribute in a perturbed dataset is $min\{\prod(X_i|X_i^p)\}_{i=1}^n$, and is called the minimum privacy guarantee. The attack resistance minimum guarantee is measured by testing the strength of the perturbed instances against the corresponding pool of data reconstruction attacks. For this task, each data reconstruction attack will reconstruct a dataset by attacking the perturbed data instance. Each reconstructed dataset is compared with the original dataset to produce n number of $Var(P)$ values, where P represents the difference between an original attribute and its reconstructed attribute, and n is the number of attributes. From the n number of $Var(P)$ values the minimum $Var(P)$ ($Var(P)^{min}$) is selected to represent the most vulnerable attribute under the corresponding attack. From all the reconstructed instances, the minimum $Var(P)^{min}$ ($Var(P)_{min}$) is selected to represent the overall vulnerability of the corresponding perturbed instance under the given set of attacks. The utility is the accuracy measured under the corresponding set of applications in the application pool. For example, when the application is data classification, PPaaS considers classification accuracy as the utility measurement. PPaaS selects the privacy preservation approach or the perturbed dataset that returns the highest FI, which represents the case with the best balance between privacy and utility.

We ran experiments with PPaaS using five different datasets, five different classification algorithms, and four different privacy-preservation algorithms that are benchmarked to produce good utility over the corresponding classification algorithms. Our experiments show that the four privacy preservation algorithms are ranked differently based on the application and the input dataset. The highest values of FI indicate the highest privacy, attack resistance, and utility with the best balance between them. After comparing the FI values (available in Table 4) generated using the values available in Table 2, we can conclude that FI provides high values, if and only if all utility, privacy, and attack resistance returned by the corresponding method are high. In all other cases, the fuzzy inference system (FIS) produces lower values for the FI. Hence, FI enables PPaaS to identify the best-perturbed dataset generated by the most suitable privacy preservation algorithm for the corresponding pool of algorithms and for the corresponding input dataset. As described in the introduction (refer to Section 1), selecting the best perturbation approach has to consider several aspects; this research focuses on privacy, attack resistance and utility, and performance is of secondary importance. By imposing a limit on execution time, PPaaS still ensures that computations will be completed in finite time, as shown in Section 5.

7. Conclusion

This paper introduced a novel framework named Privacy Preservation as a Service (PPaaS), which tailors privacy preservation to stakeholders' needs. PPaaS reduces the complexity of choosing the best data perturbation algorithm from a large number of privacy preservation algorithms. The ability to apply the best perturbation while preserving enough utility makes PPaaS an excellent solution for big data perturbation. In order to select the best privacy preservation method, PPaaS uses a fuzzy inference system (FIS) that enables PPaaS to generate ranks that are expressed as fuzzy indices for the privacy preservation algorithms applied to a dataset for a given application. The experimental results show that the fuzzy indices are good indicators of a particular privacy preservation algorithm's ability to maintain a good balance between privacy and utility.

References

[1] M. Chamikara, P. Bertok, D. Liu, S. Camtepe, I. Khalil, Efficient data perturbation for privacy preserving and accurate data stream mining, Pervasive and Mobile Computing 48 (2018) 1–19.

[2] M. Chamikara, P. Bertok, D. Liu, S. Camtepe, I. Khalil, Efficient privacy preservation of big data for accurate data mining, Information Sciences.

[3] V. Torra, Fuzzy microaggregation for the transparency principle, Journal of Applied Logic 23 (2017) 70–80. doi:https://doi.org/10.1016/j.jal.2016.11.007.

[4] A. Hasan, Q. Jiang, J. Luo, C. Li, L. Chen, An effective value swapping method for privacy preserving data publishing, Security and Communication Networks 9 (16) (2016) 3219–3228. doi:https://doi.org/10.1002/sec.1527.

[5] Y. A. A. S. Aldeen, M. Salleh, M. A. Razzaque, A comprehensive review on privacy preserving data mining, SpringerPlus 4 (1) (2015) 694. doi:https://doi.org/10.1186/s40064-015-1481-x.

[6] B. D. Okkalioglu, M. Okkalioglu, M. Koc, H. Polat, A survey: deriving private information from perturbed data, Artificial Intelligence Review 44 (4) (2015) 547–569. doi:https://doi.org/10.1007/s10462-015-9439-5.

[7] C. Dwork, A. Roth, et al., The algorithmic foundations of differential privacy, Foundations and Trends® in Theoretical Computer Science 9 (3–4) (2014) 211–407. doi:http://dx.doi.org/10.1561/0400000042.

[8] P. C. M. Arachchige, P. Bertok, I. Khalil, D. Liu, S. Camtepe, M. Atiquzzaman, Local differential privacy for deep learning, IEEE Internet of Things Journal.

[9] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, L. Zhang, Deep learning with differential privacy, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2016, pp. 308–318.

[10] R. Shokri, V. Shmatikov, Privacy-preserving deep learning, in: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, ACM, 2015, pp. 1310–1321.

[11] J. Zhao, Y. Chen, W. Zhang, Differential privacy preservation in deep learning: Challenges, opportunities and solutions, IEEE Access 7 (2019) 48901–48911.

[12] A. Zigomitros, F. Casino, A. Solanas, C. Patsakis, A survey on privacy properties for data publishing of relational data, IEEE Access 8 (2020) 51071–51099.

[13] C. C. Aggarwal, Privacy-preserving data mining, in: Data Mining, Springer, 2015, pp. 663–693.

[14] L. Xu, C. Jiang, Y. Chen, Y. Ren, K. R. Liu, Privacy or utility in data collection? a contract theoretic approach, IEEE Journal of Selected Topics in Signal Processing 9 (7) (2015) 1256–1269.

[15] R. L. Wilson, P. A. Rosen, Protecting data through'perturbation'techniques: The impact on knowledge discovery in databases, in: Information Security and Ethics: Concepts, Methodologies, Tools, and Applications, IGI Global, 2008, pp. 1550–1561.

[16] R. Agrawal, R. Srikant, Privacy-preserving data mining, in: ACM Sigmod Record, Vol. 29, ACM, 2000, pp. 439–450. doi:https://doi.org/10.1145/335191.335438.

[17] E. Bertino, I. N. Fovino, L. P. Provenza, A framework for evaluating privacy preserving data mining algorithms, Data Mining and Knowledge Discovery 11 (2) (2005) 121–154.

[18] K. Chen, L. Liu, A random rotation perturbation approach to privacy preserving data classification, The Ohio Center of Excellence in Knowledge-Enabled Computing.
URL https://corescholar.libraries.wright.edu/knoesis/916/

[19] K. Chen, L. Liu, Geometric data perturbation for privacy preserving outsourced data mining, Knowledge and Information Systems 29 (3) (2011) 657–695. doi:https://doi.org/10.1007/s10115-010-0362-4.

[20] D. Bogdanov, S. Laur, J. Willemson, Sharemind: A framework for fast privacy-preserving computations, Computer Security-ESORICS 2008 (2008) 192–206.

[21] S. Agrawal, J. R. Haritsa, A framework for high-accuracy privacy-preserving mining, in: Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on, IEEE, 2005, pp. 193–204.

[22] B. Thuraisingham, M. Kantarcioglu, E. Bertino, C. Clifton, Towards a framework for developing cyber privacy metrics: A vision paper, in: Big Data (BigData Congress), 2017 IEEE International Congress on, IEEE, 2017, pp. 256–265.

[23] P. Kairouz, S. Oh, P. Viswanath, Extremal mechanisms for local differential privacy, in: Advances in neural information processing systems, 2014, pp. 2879–2887.

[24] K. Muralidhar, R. Parsa, R. Sarathy, A general additive data perturbation method for database security, management science 45 (10) (1999) 1399–1415.

[25] A. Hundepool, J. Domingo-Ferrer, L. Franconi, S. Giessing, E. S. Nordholt, K. Spicer, P.-P. De Wolf, Statistical disclosure control, John Wiley & Sons, 2012.

[26] S. Martínez, D. Sánchez, A. Valls, Towards k-anonymous non-numerical data via semantic resampling, in: International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Springer, 2012, pp. 519–528.

[27] C. C. Aggarwal, P. S. Yu, A condensation approach to privacy preserving data mining, in: EDBT, Vol. 4, Springer, 2004, pp. 183–199.

[28] K. Liu, H. Kargupta, J. Ryan, Random projection-based multiplicative data perturbation for privacy preserving distributed data mining, IEEE Transactions on knowledge and Data Engineering 18 (1) (2006) 92–106.

[29] C. C. Aggarwal, P. S. Yu, On privacy-preservation of text and sparse binary data with sketches, in: Proceedings of the 2007 SIAM International Conference on Data Mining, SIAM, 2007, pp. 57–67.

[30] A. Jones, K. Leahy, M. Hale, Towards differential privacy for symbolic systems, in: 2019 American Control Conference (ACC), IEEE, 2019, pp. 372–377.

[31] A. Machanavajjhala, D. Kifer, Designing statistical privacy for your data, Communications of the ACM 58 (3) (2015) 58–67.

[32] M. A. P. Chamikara, P. Bertok, D. Liu, S. Camtepe, I. Khalil, Efficient privacy preservation of big data for accurate data mining, Information Sciences, Elsevier 527 (2019) 420–443. doi:10.1016/j.ins.2019.05.053.

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

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

[35] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam, l-diversity: Privacy beyond k-anonymity, ACM Transactions on Knowledge Discovery from Data (TKDD) 1 (1) (2007) 3–es.

[36] L. Zhang, S. Jajodia, A. Brodsky, Information disclosure under realistic assumptions: Privacy versus optimality, in: Proceedings of the 14th ACM conference on Computer and communications security, ACM, 2007, pp. 573–583.

[37] S. R. Ganta, S. P. Kasiviswanathan, A. Smith, Composition attacks and auxiliary information in data privacy, in: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2008, pp. 265–273.

[38] R. C.-W. Wong, A. W.-C. Fu, K. Wang, P. S. Yu, J. Pei, Can the utility of anonymized data be used for privacy breaches?, ACM Transactions on Knowledge Discovery from Data (TKDD) 5 (3) (2011) 16.

[39] C. Dwork, The differential privacy frontier, in: Theory of Cryptography Conference, Springer, 2009, pp. 496–502.

[40] N. Mohammed, R. Chen, B. Fung, P. S. Yu, Differentially private data release for data mining, in: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2011, pp. 493–501.

[41] W. Fan, J. He, M. Guo, P. Li, Z. Han, R. Wang, Privacy preserving classification on local differential privacy in data centers, Journal of Parallel and Distributed Computing 135 (2020) 70–82.

[42] Q. Wang, Z. Li, Q. Zou, L. Zhao, S. Wang, Deep domain adaptation with differential privacy, IEEE Transactions on Information Forensics and Security.

[43] K. Chen, G. Sun, L. Liu, Towards attack-resilient geometric data perturbation, in: proceedings of the 2007 SIAM international conference on Data mining, SIAM, 2007, pp. 78–89.

[44] K. Liu, C. Giannella, H. Kargupta, A survey of attack techniques on privacy-preserving data perturbation methods, in: Privacy-Preserving Data Mining, Springer, 2008, pp. 359–381.

[45] Y. Gupta, A. Saini, A. Saxena, A new fuzzy logic based ranking function for efficient information retrieval system, Expert Systems with Applications 42 (3) (2015) 1223–1234.

[46] V. X. Tran, H. Tsuji, Qos based ranking for web services: Fuzzy approaches, in: 2008 4th International Conference on Next Generation Web Services Practices, IEEE, 2008, pp. 77–82.

[47] T. Drabas, D. Lee, Learning PySpark, Packt Publishing Ltd, 2017.

[48] K. Chen, L. Liu, Privacy preserving data classification with rotation perturbation, in: Data Mining, Fifth IEEE International Conference on, IEEE, 2005, pp. 4–pp.

[49] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436–444.

[50] D. Agrawal, C. C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2001, pp. 247–255.

[51] M. Sokolova, N. Japkowicz, S. Szpakowicz, Beyond accuracy, f-score and roc: a family of discriminant measures for performance evaluation, in: Australasian joint conference on artificial intelligence, Springer, 2006, pp. 1015–1021.

[52] I. H. Witten, E. Frank, M. A. Hall, C. J. Pal, Data Mining: Practical machine learning tools and techniques, Morgan Kaufmann, 2016.
URL `https://books.google.com.au/books?isbn=0128043571`