PriSM: A Privacy-friendly Support vector Machine

Michele Barbato[0000−0002−8521−896X], Alberto Ceselli[0000−0002−0983−2706],
Sabrina De Capitani di Vimercati[0000−0003−0793−3551],
Sara Foresti[0000−0002−1658−6734], and Pierangela Samarati[0000−0001−7395−4620]

Università degli Studi di Milano - 20133, Milano, Italy
firstname.lastname@unimi.it

Abstract. Today's society is witnessing not only an evergrowing dependency on data, but also an increasingly pervasiveness of related analytics and machine learning applications. From business to leisure, the availability of services providing answers to questions brings great benefits in diverse domains. On the other side of the coin, the need to provide input data that the services need to compute a response. However, some data may be considered sensitive or confidential and users would legitimately be reluctant to release them to third parties.
Considering classification tasks in machine learning applications, we introduce our PriSM (Privacy-friendly Support vector Machine) approach for computing a privacy-friendly model. PriSM anticipates the training phase of the classifier with a phase for discovering correlations among attributes that can indirectly expose sensitive information. It then trains the classifier excluding from consideration not only sensitive attributes but also other sets of attributes that have been learned as correlated to them. The result is a privacy-friendly classifier that does not require any of such information as input from the users. Our experimental evaluation on both synthetic and real-world datasets confirms the effectiveness of PriSM in protecting privacy while maintaining classification accuracy.

Keywords: PriSM, privacy-friendly classifier, sensitive attribute, sensitive correlation

1 Introduction

In machine learning, data classification is a method where a model tries to predict the correct label of a given input data. The model learns to predict labels during a *training* phase, where a statistical relationship between attribute values and labels is identified by analyzing a large number of samples with known attributes and labels. After training the model (classifier) for *prediction*, with the attribute values of a user as input the model generates a corresponding label as output. Clearly, the more the data available the more accurate the classification, and a data-hungry approach would try to employ all available attributes for classification, which in turn will require users of the classifier application to provide input for all such attributes.

In this paper, we consider the problem of building, from a training dataset, a classifier that can be made available to third parties and that end users can use for classifying their data. In this context, some attributes may be considered sensitive (or company-confidential for business application scenarios) and users of the application would not be willing to disclose them, hence a privacy-friendly classifier should not assume their availability and therefore sensitive attributes should also not be used in the training phase so that classification does not depend on them. As a motivating example, consider a medical center that has information about patients and aims at releasing a classifier that people can use at home to have suggestions on suitable physical activities to improve their fitness. The classifier is not run by the medical center but offered through an external provider. The set of attributes in the medical dataset includes various information about the patients. While some of the attributes are not considered sensitive (e.g., age), others are to be considered sensitive (e.g., a disease) and their values should remain confidential to the provider (i.e., the classifier should not require them). If the classification model does not depend on an attribute (e.g., either because it is irrelevant or because it has been artificially excluded), the corresponding value is not required for performing the prediction on the label. Note that, excluding sensitive information from the training process not only enables producing a classifier that does not require it for prediction but also ensures that the model released to the external provider does not leak sensitive information from the training data. A naive approach to enforce such protection would be to simply discard sensitive attributes from the dataset before training and hence ignore them all throughout. However, such an approach would still be exposed to improper sensitive information leakage through data dependencies and correlations. As a matter of fact, values of the sensitive attributes may be indirectly leaked by other attributes that - individually or in combination - can convey information on sensitive attributes. For instance, a disease (sensitive attribute) may be indirectly exposed by the values of medicine prescriptions (the cure) or by a combination of values of some physical parameters. While some data dependencies, such as the ones just mentioned, may be known, others may be hidden in the data and a truly privacy-friendly approach should ensure protection even with respect to them (in fact, the external provider servicing the application to the user can employ a classification process at their side as well).

Our approach, called PriSM (for Privacy-friendly Support vector Machine), addresses this problem by excluding from the training process sensitive attributes as well as attributes that may leak information on them. More precisely, PriSM first learns correlations of other attributes in the dataset with sensitive attributes. It then restricts the training process forcing the classifier to exclude from consideration not only the sensitive attributes but also sets of attributes that can leak them. It does so while, at the same time, minimizing the effect of such protection on the correctness of the classification. Even more, PriSM accounts for the fact that what can be considered sensitive are, in some cases, only specific values. For instance, while a disease like `flu` may be considered non problematic, values of other rarer or discriminatory diseases need strong

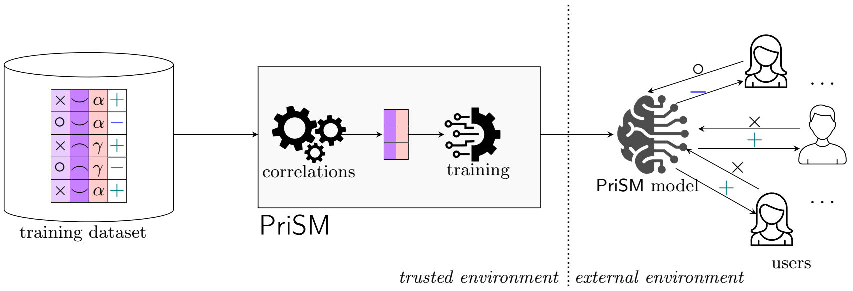


Fig. 1: Reference scenario

protection. In learning correlations for attributes for which only some values are critical, we specifically address correlations with respect to such critical values so to find those correlations that are problematic (in contrast to any correlation). This notwithstanding the fact that the sensitive attribute is to be excluded in its entirety in training the classifier.

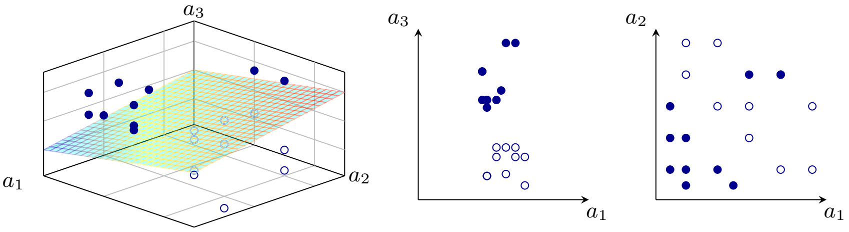
Figure 1 illustrates the overall scenario of our approach. The training phase is performed within the trusted environment under the data owner control, and hence with visibility of the whole dataset. The result is a classifier to be released to an external environment and made available to users. Users can input their data and receive a response as predicted by the classifier.

The remainder of the paper is organized as follows. Section 2 presents the main concepts and the formulation of the problem introducing our PriSM approach to compute a privacy-friendly classifier. Sections 3 and 4 describe the two phases of PriSM, discovering correlations among attributes in the dataset and then training the classifier to exclude from consideration sensitive attributes as well as other sets of attributes that have been learned as correlated to them. Section 5 presents our experimental evaluation confirming the effectiveness of PriSM in protecting privacy while minimizing the impact on the quality of the classification. Section 6 discusses related works. Finally, Section 7 presents our conclusions. Appendix A reports theorems proving the correctness of PriSM.

2 PriSM

For concreteness, we assume a support vector machine as a classifier. We note, however, that our approach can be extended to more general classification problems and to other data analytics tasks (e.g., regression tasks). Also, we assume a single sensitive attribute (which can be sensitive in its entirety or for which only some values may be defined as critical), and a binary classification problem, that is, the classification (*label*) attribute has domain in $\{+1, -1\}$.

The training dataset is modeled as a relational table r defined over schema $R(A, s, l)$, where $A = \{a_1, \ldots, a_n\}$ is a set of attributes other than the sensitive attribute s and the label attribute l. Training a classifier on r for predicting l



(a) $P = \{a_1, a_2, a_3\}$ **(b)** $P = \{a_1, a_3\}$ **(c)** $P = \{a_1, a_2\}$

Fig. 2: An example of linear classifier in a 3-dimensional space (a) and of the projection of points in the 3D space over two bi-dimensional spaces (b)-(c)

means learning from the tuples in r how the value of l can be predicted based on the values of the other attributes in a tuple (i.e., learning the relationships between the values of the attributes in $R \setminus \{l\}$ and known labels in a training dataset). Assuming to represent the projection of tuples in r over $R \setminus \{l\}$ as points in a $|R \setminus \{l\}|$-dimensional space, the training phase of a support vector machine classifier finds a hyperplane that well separates the points with positive $(+1)$ from the points with negative (-1) label, corresponding to the two classes. The predicted label will depend on the side of the hyperplane where the point representing the user's data falls in the multi-dimensional space. For instance, Figure 2(a) illustrates a hyperplane in a 3-dimensional space that well separates positive (filled circles) from negative (empty circles) points.

Our goal is to ensure the classifier to be *privacy-friendly*, and perform predictions based only on a subset P of attributes in $R \setminus \{l\}$ that does not include the sensitive attribute nor other attributes that can leak its values (or those values specified as critical for it).

PriSM works in two phases: the first phase identifies sensitive correlations of other attributes with the sensitive attribute; the second phase trains the classifier in a controlled way to restrict the choice of predictor attributes so that privacy is respected, while minimizing the impact on the quality of the classifier. More precisely, the first phase of PriSM learns from the training dataset the correlations among the other attributes A and the sensitive attribute s, meaning the sets of attributes that can well predict the sensitive attribute (we will elaborate more on this in Section 3). We note that if a set X of attributes is correlated with s, denoted $X \rightsquigarrow s$, so it is clearly any set $Y \supset X$. Also, blocking an inference channel from X to s, forbidding the classifier to use all the attributes in X as predictor attributes, trivially blocks the inference channel from any $Y \supset X$ to s. We are therefore interested in the identification of a set of minimal sensitive correlations for r, as formally captured by the following definition.

Definition 1 (Set of minimal sensitive correlations). *Let $R(A, s, l)$ be a relation schema, with s the sensitive attribute and l the label attribute. The set of* minimal sensitive correlations *for s is a set $\mathcal{X}$ of subsets of A such that: 1)*

$\forall X \in \mathcal{X} : X \rightsquigarrow s$; 2) $\forall X_i \rightsquigarrow s : \exists X_j \in \mathcal{X}$ s.t. $X_j \subseteq X_i$; 3) $\forall X_i, X_j \in \mathcal{X}, i \neq j :$ $X_i \not\subset X_j$ and $X_j \not\subset X_i$.

In the definition, the first two conditions ensure that $\mathcal{X}$ includes only existing correlations (Condition 1) and that all correlations are captured (Condition 2). Condition 3 ensures that only minimal correlations are explicitly represented. Besides correlations learned from the training dataset, $\mathcal{X}$ can also include additional (minimal) correlations specified by the data owner [8]. Given a set of minimal sensitive correlations for a relation R, a classifier is said to be privacy-friendly if the set of attributes used for classification does not include the sensitive attribute s nor (in its entirety) any set X of attributes correlated to it. This concept is formally captured by the following definition of privacy-friendly classifier.

Definition 2 (Privacy-friendly classifier). *Let $R(A, s, l)$ be a relation schema, with s the sensitive attribute and l the label attribute. A classifier C for l using as predictor attributes a set $P \subseteq R \setminus \{l\}$ of attributes is* privacy-friendly *iff: $s \notin P$ and $\forall X \subseteq P, X \not\rightsquigarrow s$.*

In other words, a classifier is privacy-friendly if the set P of attributes that it uses as predictor attributes does not include any set in $\mathcal{X}$ (formally, $\forall X \in \mathcal{X}$: $X \nsubseteq P$). Note that for a set X of attributes not to be included in P it is sufficient to exclude one (any) of its attributes. In fact, while on one hand excluding more attributes can clearly increase privacy (the less the data, the less the potential inference on the sensitive attribute) on the other hand removing more attributes than needed (e.g., at the extreme, all the attributes involved in a correlation) can affect severely the ability of the classifier to predict the value of the label attribute, destroying any utility of the classifier. Since the set of correlations is minimal, such an aggressive exclusion is not needed, as ensuring exclusion of one attribute for each of the correlations would have the same effect. Additionally, the same attribute can solve more than one correlation.

Clearly, there can be different choices for a set of predictors that satisfy Definition 2, each with a different impact on the quality of the classifier, with the usual dichotomy between privacy and utility. The challenge is to find a set P of attributes that ensures privacy while minimizing the effect on the prediction quality of the classifier. As an example, consider Figure 2(a), where X=$\{a_1$,a_2,$a_3\}$ is a sensitive correlation. If the set of selected predictor attributes is P=$\{a_1$,$a_3\}$ (Figure 2(b)), it is possible to find a linear classifier that well separates the positive (filled circles) from the negative (empty circles) points. By contrast, using P=$\{a_1$,$a_2\}$ (Figure 2(c)) would imply a higher misclassification since the positive and negative points are not linearly separable.

Capturing the impact on the quality of a classifier C in terms of misclassification [16], denoted $\epsilon(C)$, our problem is formalized as follows.

Problem 1 *Given a training dataset r defined over relation schema $R(A, s, l)$, with s the sensitive attribute and l the label attribute, find a classifier C that is privacy-friendly (Definition 2) and that minimizes misclassification. That is, there is no classifier C' satisfying Definition 2 such that $\epsilon(C') \leq \epsilon(C)$.*

PriSM solves the problem in two phases, first learning from the training dataset the set of minimal correlations representing inference channels to the sensitive attribute (Section 3), and then performing training of the classifier controlling and restricting the choice of the set of predictor attributes (Section 4).

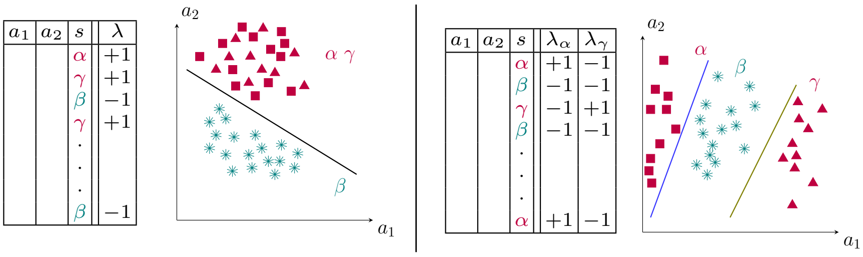
3 Sensitive Correlations Discovery

The first phase of our approach aims at learning sensitive correlations from the training dataset, that is, correlations that may leak (critical) values of the sensitive attribute. More formally, the first phase aims to find the set $\mathcal{X}=\{X_1,\ldots,X_n\}$ of all minimal sensitive correlations $X_i \rightsquigarrow s$, $i = 1,\ldots,n$. In the following, we first discuss how to determine whether, for a given set $X \subseteq A$ of attributes, $X \rightsquigarrow s$ holds (Section 3.1), and then how to identify the set of candidates X against which a correlation needs to be evaluated (Section 3.2).

3.1 Assessing Correlations

A set X of attributes is correlated with s if X is a good predictor for the values of s. Intuitively, correlation between X and s can be evaluated running a classifier (i.e., treating s as the label) and comparing some metrics on the classifier result with a threshold value τ (reflecting the accuracy of the prediction). Correlation exists whenever the metrics evaluates above the threshold.

As already mentioned, our approach for discovering such correlations also accounts for situations in which only specific values of the sensitive attribute are considered critical. Notwithstanding the fact that the sensitive attribute is to be discarded in its entirety in training (and prediction), considering those values that are critical (if not all are) for s in discovering correlations enables to be more precise in spotting those correlations that lead to a critical value, and not just any value of the sensitive attribute. The identification of correlations for the case where not all values of the sensitive attribute are critical deserves some considerations. In particular, a question to solve is whether correlations should be identified considering the set of critical values as a whole (single encoding) or considering each critical value individually (multiple encoding). Intuitively, considering the critical values of the sensitive attribute as a whole equates to consider a binary label λ for s, with $\lambda = +1$ for all tuples t such that $t[s]$ is critical; $\lambda = -1$, otherwise. This enables to detect the set of attributes that are correlated to the whole set of critical values. By contrast, considering the different sensitive values separately implies considering a binary label λ_v for each critical value v with $\lambda_v = +1$ for all tuples t such that $t[s]=v$; $\lambda_v = -1$, otherwise. We note that neither of the two approaches subsumes the other since each of them can discover sensitive correlations that would go undetected from the other. To illustrate, consider two datasets where $A=\{a_1,a_2\}$ and, among the values of the sensitive attribute, only α and γ are critical. Figure 3 illustrates the tuples in these two datasets as points in a bi-dimensional space where the dimensions correspond to a_1 and a_2. In the bi-dimensional space, α and γ are denoted with



(a) single encoding **(b)** multiple encoding

Fig. 3: An example of two datasets with single (a) or multiple encoding (b) together with their multidimensional representation

a red square (for α) and a red triangle (for γ) while a green star denotes any other value. Suppose that we aim at discovering whether there is a correlation between $X{=}\{a_1{,}a_2\}$ and the sensitive attribute s. For the first dataset, the critical values taken all together are linearly separable from the non-critical ones, while each critical value singularly taken is not linearly separable from all the other values (critical or not critical). Therefore, single encoding finds the correlation between values in X and s while multiple encoding would not. On the contrary, for the second dataset, the critical values taken all together are not linearly separable from the non-critical ones, while each critical value singularly taken is linearly separable from all the other values (critical or not critical). Therefore, multiple encoding finds the correlation between values in X and s while single encoding would not.

We further note that, while in principle correlations can be learned by running a classifier (e.g., training an SVM having non-zero value only for the coefficients of the hyperplane of the attributes in the set X to be evaluated), this would be quite computationally intensive. We use instead correlation coefficients [16] as a proxy for such evaluation. The specific correlation coefficient, or combination of them, that can perform better may depend on the specific dataset. In preliminary experiments we compared many options, finding Pearson's correlation coefficient and Cramer's V test of statistical independence to be a good choice in general.

3.2 Finding Potential Candidates

To identify the candidate sets of attributes against which correlations need to be evaluated, we leverage the natural monotonicity of sensitive correlations similarly to Apriori strategy for frequent itemset mining [1], that is, if $X{\rightsquigarrow}s$, then $X'{\rightsquigarrow}s$ $\forall X' \subset R$ such that $X' \supset X$. We do so by performing different iterations over variable c corresponding to the cardinality of X (e.g., $c = 1$ for evaluating individual attributes at the first iteration), with each iteration determining the set $\mathcal{Y}_c$ of candidates of cardinality c. Differently from Apriori, which generates candidates for iteration c by joining pairs of sets from iteration $c-1$ that have

```
INPUT    dataset $r$ on $R(A,s,l)$
         sensitive attribute $s$ (and critical values)
         maximum cardinality of the set of predictor attributes $pmax$
         threshold correlation $\tau$
OUTPUT set $\mathcal{X}$ of minimal sensitive correlations for $s$

MAIN
1:  $\mathcal{X} := \emptyset$
2:  $c := 0$
3:  repeat
4:      $c := c + 1$
5:      if $c = 1$
6:      then let $\mathcal{Y}_1$ be the set of all sets $\{a\}$ s.t. $a \in A$
7:      else  let $\mathcal{Y}_c$ be the set of all sets $Y$ with $|Y| = c$
                s.t. $Y = \bigcup_i X_i \in Q$, with $Q \subseteq \mathcal{Y}_{c-1}$, $|Q| = c$
8:      for each $Y \in \mathcal{Y}_c$ do
9:        if correlation($Y$,$s$)$\geq \tau$ /* correlation with single and/or multiple encoding */
10:       then $\mathcal{Y}_c := \mathcal{Y}_c \setminus \{Y\}$
11:             $\mathcal{X} := \mathcal{X} \cup \{Y\}$
12: until $c = pmax$ OR $|\mathcal{Y}_c| \leq c$
```

Fig. 4: Procedure computing sensitive correlations

the first $c-2$ attributes in common, our algorithm considers as candidates for iteration c only those sets produced by the union of c sets from iteration $c-1$. For instance, assume at the end of iteration 2, $\mathcal{Y}_2$ to include $\{a_i, a_j\}$ and $\{a_i, a_z\}$, but not $\{a_j, a_z\}$, joining the two sets would produce as candidate to consider $\{a_i, a_j, a_z\}$. However, such a set cannot belong to $\mathcal{X}$, since one of its subsets already does and $\mathcal{X}$ is a minimal set (Definition 1). In fact, $\{a_j, a_z\} \notin \mathcal{Y}_2$ implies that at least one between $\{a_j\} \in \mathcal{X}$ and $\{a_z\} \in \mathcal{X}$ holds. Our construction avoids producing such candidates that would then be discarded for minimality, ensuring to produce all and only candidates that are not a superset of sets already included in $\mathcal{X}$ in a previous iteration (see Theorem 2 in the Appendix). The algorithm assumes also a limit *pmax* on the number of attributes in candidate sensitive correlations, corresponding to the maximum number of predictor attributes used by PriSM (see Section 4). The reason for such a limit is both efficiency, to limit the iterations, as well as supporting a principle of *parsimony* for the number of attributes to be used as predictors and hence requested as input to the users of the application (clearly *pmax*=$|A|$ implies no limitation).

Figure 4 illustrates the procedure for computing the set of sensitive correlations. Starting from the set $\mathcal{Y}_1$ of candidate correlations including one attribute only (line 6), for each Y in $\mathcal{Y}_1$, the algorithm verifies if $Y \rightsquigarrow s$ holds and, if this is the case, it removes Y from $\mathcal{Y}_1$ inserting Y into $\mathcal{X}$ (lines 9-11). After evaluating all the singleton sets, $\mathcal{Y}_1$ will include only those attributes a such that $\{a\} \not\rightsquigarrow s$. The algorithm then checks all the pairs in $\mathcal{Y}_2$ composed of attributes in $\mathcal{Y}_1$, because for any other pair X of attributes there exists an attribute $a{\in}X$ that has been removed from $\mathcal{Y}_1$ (and included in $\mathcal{X}$), therefore X does not need to be inserted into $\mathcal{X}$. The algorithm evaluates candidate sets Y of increasing size c. For each value of c, the check is limited to the candidate correlations Y including c attributes, obtained as the union of c sets in $\mathcal{Y}_{c-1}$ (line 7). This restricts $\mathcal{Y}_c$ to the sets Y such that all the subsets of Y of cardinality $c-1$ belong to $\mathcal{Y}_{c-1}$. Indeed, if at least a subset X of Y does not appear in $\mathcal{Y}_{c-1}$, it means that a correlation $X{\rightsquigarrow}s$ dominating $Y{\rightsquigarrow}s$ has already been learned by the algorithm. The algorithm stops when c reaches *pmax* attributes (larger correlations would for sure not be exposed by a classifier using at most *pmax* predictors), or when the candidate set of correlations including c attributes has less than $c+1$ subsets of attributes (any set of $c+1$ attributes would not have all its subsets of c attributes in $\mathcal{Y}_c$, hence $\mathcal{Y}_{c+1}$ would be empty).

The output of the first phase is then the set $\mathcal{X}$ of minimal correlations among attributes A in the dataset and the sensitive attribute. Note that this phase does not enforce any choice (or removal) of attributes from the classifier. It is for the second phase (next section) to determine the optimal set of attributes that does not include in its entirety any set in $\mathcal{X}$ and optimizes label prediction.

4 Classifier Training

The second phase of our approach consists in simultaneously selecting the set of attributes and training a classifier, thus solving Problem 1. While the problem applies to a general classification task, and the first phase (Section 3) is agnostic with respect to the classifier, the execution of this second phase depends on the classifier to be considered. As already noted in the previous sections, we consider classification with a Support Vector Machine (SVM).

We then design a variant of SVM that controls and restricts predictor attributes selection. In the following, we use boldface for denoting vectors. Similarly to classical SVMs, each tuple t in the training dataset r is modeled as a point in a multi-dimensional space, having a dimension for each attribute in $R\setminus\{l\}$. The classification model is geometrically represented as a hyperplane $\mathcal{H}$ in the multidimensional space. Training a SVM then corresponds to learn the coefficients $\mathbf{w}{\in}\mathbb{R}^{|R|-1}$ and $b{\in}\mathbb{R}$ of a hyperplane $\mathcal{H}{=}\ \{\mathbf{t}\in\mathbb{R}^{|R|-1}\colon \mathbf{w}\cdot\mathbf{t}=b\}$, with $\mathbf{t}$ being the vector of values of tuple $t[R\setminus\{l\}]{\in}r$. The hyperplane must separate well (i.e., place on different sides) points with positive ($t[l]=+1$) and negative ($t[l]=-1$) label in the training dataset. Each coefficient $\mathbf{w}[a]$, with $a\in R\setminus\{l\}$, used in the definition of $\mathcal{H}$ represents the slope of the hyperplane in the dimension that corresponds to attribute a. Since finding a hyperplane that separates positive from negative points might not always be possible (e.g., when the positive and negative classes are not linearly separable), our training phase relies on soft margin [6]. Intuitively, it considers a misclassification penalty when maximizing the distance between the positive and negative classes.

Differently from standard SVMs, our problem has a combinatorial nature. A straightforward adaptation would require to enumerate all possible subsets

$$\min \frac{1}{2}||\mathbf{w}||_2^2 + \pi \sum_{i=1}^{|r|} \mathbf{z}[t_i] \tag{1}$$

$$t_i[l](\mathbf{w} \cdot \mathbf{t}_i - b) \geq 1 - \mathbf{z}[t_i] \qquad i = 1, \ldots, |r| \tag{2}$$

$$\mathbf{z}[t_i] \geq 0 \qquad i = 1, \ldots, |r| \tag{3}$$

$$\mathbf{w}[a] \in \mathbb{R}, b \in \mathbb{R} \qquad \forall a \in R \setminus \{l\} \tag{4}$$

$$\mathbf{p}[a]\mathbf{w}^{\mathbf{L}}[a] \leq \mathbf{w}[a] \leq \mathbf{p}[a]\mathbf{w}^{\mathbf{U}}[a] \qquad \forall a \in R \setminus \{l\} \tag{5}$$

$$\sum_{a \in R \setminus \{l\}} \mathbf{p}[a] \leq pmax \tag{6}$$

$$\mathbf{p}[s] = 0 \tag{7}$$

$$\sum_{a \in X} \mathbf{p}[a] \leq |X| - 1 \qquad \forall X \in \mathcal{X} \tag{8}$$

$$\mathbf{p}[a] \in \{0, 1\} \qquad \forall a \in R \setminus \{l\} \tag{9}$$

Fig. 5: Mixed Integer Programming formulation of the PriSM training problem

of predictors, testing each of them for sensitive correlations, and solve a SVM training problem for the remaining attributes. Such an approach, having a time complexity exponential in the number of predictors, would be computationally infeasible. We solve Problem 1 by extending the classical formulation of the SVM training problem as an optimization problem, formulated as a Mixed Integer Program [27] imposing *privacy-friendliness* (the predictors cannot include the sensitive attribute nor any set correlated to it) and *parsimony* (use at most *pmax* attributes as predictors) as model constraints. Figure 5 illustrates the formalization of the optimization problem, where the variables are as follows.

- $pmax \in [1, |R|)$: input parameter representing the maximum number of predictors that can be used by the classifier.
- $\pi \in \mathbb{R}$: input parameter representing the relative penalty for misclassification errors (higher values correspond to a smaller probability of misclassification, at the price of smaller separation between positive and negative classes).
- $\mathbf{w}^{\mathbf{U}} \in \mathbb{R}^{|R|-1}$, $\mathbf{w}^{\mathbf{L}} \in \mathbb{R}^{|R|-1}$: input parameters representing the upper and lower bounds on the values of $\mathbf{w}$ for each attribute in $R \setminus \{l\}$ (which represent the maximum and minimum slope of the hyperplane allowed in the corresponding direction). They forbid the degenerate choice of vertical hyperplanes, and improve the numerical stability of our optimization procedure (narrow bounds speed up convergence, looser ones reduce the risk of cutting off solutions that are potentially optimal [2]).
- $\mathbf{t}_i \in \mathbb{R}^{|R|-1}$, $t_i[l] \in \{-1, +1\}$: input vector of values for attributes in $R \setminus \{l\}$ and label, respectively, for each tuple $t_i \in r$ in the training dataset.
- $\mathbf{p} \in \{0, 1\}^{|R|-1}$: resulting binary variables modeling the selection (value 1) or exclusion (value 0) of each attribute $a \in R \setminus \{l\}$ from the set of predictors.

- $\mathbf{w}{\in}\mathbb{R}^{|R|-1}$, $b{\in}\mathbb{R}$: resulting coefficients of the hyperplane;
- $\mathbf{z}{\in}\mathbb{R}^{|r|}$: resulting misclassification error for each tuple $t{\in}r$.

Equation 1 is the classical soft-margin optimization function used in training SVM classifiers, and Equations 2-4 correspond to classical constraints of the definition of the SVM, when formulated as a Mixed Integer Linear Program. The additional constraints (Equations 5-9), enforce instead restrictions which are specific for our PriSM problem. The semantics of the constraints is as follows.

(2) models the possible error $\mathbf{z}$ in classification using the trained classifier. More precisely, $(\mathbf{w} \cdot \mathbf{t}_i - b)$ has a positive value if the predicted label for t_i is $+1$; it has a negative value if the predicted label is -1. The product between $(\mathbf{w}{\cdot}\mathbf{t}_i{-}b)$ and the correct label $t_i[l]$ is positive if the predicted class is correct; it is negative otherwise. Therefore, $\mathbf{z}[t_i]$ is 0 if the prediction is correct (it cannot be a negative number, see (3)); it has a positive value measuring the misclassification error (i.e., the distance from the hyperplane), otherwise.

(3) limits the values of $\mathbf{z}$ to be non-negative numbers. This ensures to properly consider the penalty of misclassification in the objective function (i.e., to prevent positive and negative misclassifications that compensate each other).

(4) specifies the domain of coefficients $\mathbf{w}$ and b to be $\mathbb{R}$.

(5) constrains the values of the slope coefficient $\mathbf{w}[a]$ to be in the range $[\mathbf{w^L}[a]{,}\mathbf{w^U}[a]]$ for each attribute $a{\in}R{\setminus}\{l\}$ selected as predictor (i.e., $\mathbf{p}[a]{=}1$); at the same time it forces $\mathbf{w}[a]{=}0$ when the attribute is not selected (i.e., $\mathbf{p}[a]{=}0$).

(6) limits the number of predictors to be at most *pmax* by setting the sum for binary variables $\mathbf{p}[a]$, with $a{\in}R \setminus \{l\}$, to be lower than or equal to *pmax*.

(7) excludes the sensitive attribute from the set of predictors by setting $\mathbf{p}[s]$ to 0. Note that this also implies constraining $\mathbf{w}[s]$ to be equal to 0 (see (5)).

(8) imposes the number of attributes included in the set of predictors from each set $X{\in}\mathcal{X}$ to be smaller than the cardinality of X. This ensures that, for each X, at least one attribute is excluded from the set of predictors. The constraint forces the solution to have at least one of the binary variables $\mathbf{p}[a]$, with $a{\in}X$, set to 0 thus making their sum lower than the cardinality of X. Like for s, $\mathbf{p}[a] = 0$ implies $\mathbf{w}[a] = 0$, meaning the attribute is excluded from consideration.

(9) restricts the domain of $\mathbf{p}$ to be {0,1} (0 being exclusion, and 1 inclusion).

Intuitively, the objective function (Equation 1) aims at balancing two needs: *i)* maximize separation between the positive and the negative class, and *ii)* minimize misclassifications. To maximize separation between classes, the SVM maximizes the distance from the hyperplane of the nearest positive point and the nearest negative point in the training dataset. This is guaranteed by the first term in the objective function. The second term instead represents the misclassification penalty, obtained by multiplying the overall misclassification error of tuples in the training set by coefficient π. PriSM then offers a global optimality guarantee to choose a given number of predictors containing no sensitive correlations, minimizing misclassification error (see Theorem 2 in the Appendix). We note that, in general, finding effective formulations of hard problems as ours is far from trivial [2]. Our model, however, enjoys two properties that permit to keep training times under control: *i)* the number of binary variables is linear in the number of predictors, independently from the number of tuples in the dataset; *ii)* when integrality conditions on these (few) binary variables are relaxed, we obtain a convex quadratic model, which allows for effective resolution algorithms. This is confirmed by our experimental results, where training times were always of few seconds with about one minute at most on datasets with 16 attributes and more than 40,000 tuples.

5 Experimental Results

We performed a series of experiments to assess the effectiveness of PriSM in training a classifier that provides high accuracy in the prediction of the label attribute, without revealing critical values of the sensitive attribute.

5.1 Experimental Setting and Datasets

We implemented PriSM in `python3`, using the `python` API of `Gurobi 10.1` to solve the Mixed Integer Programming formulation of PriSM in Figure 5 with a branch-and-cut algorithm [27]. We assumed correlations of at most 5 attributes each, and used as correlation coefficient the Cramer's V test with p-value 0.05 and threshold τ to either 0.02 or 0.2 (depending on the dataset). These thresholds ensure that no subset of predictors can be selected, unless it is independent from the sensitive attribute with very high probability or its association strength is statistically very low. Based on preliminary testing, we set $\mathbf{w}^{\mathbf{U}}[a] = 1000$ and $\mathbf{w}^{\mathbf{L}}[a] = -1000$ for all attributes in $R\backslash\{l\}$. These values are large enough for not affecting optimality while enhancing computational performance [2]. The results reported in the following have been obtained using a PC equipped with an Intel Core i5-1135G7 at 2.40 GHz and 32GB of memory.

For assessing the effectiveness of PriSM, we considered a synthetic dataset and a real-world dataset. We generated the synthetic dataset as a stress-test of PriSM to enable us to control all the features that might affect our approach.[1] It contains 40,000 tuples and is defined over a relational schema with 15 attributes, including 13 candidate predictors defined over a ternary domain, a binary sensitive attribute, and a binary label attribute. The frequency distributions of the two values of the sensitive attribute and of the two values of the label attribute are balanced (52% and 50% occurrences of positive values, respectively) and all the attributes have an impact on the classification task. We selected, as real-world dataset, the binarized version of *Bank Marketing* dataset[2] that

[1] https://doi.org/10.13130/RD_UNIMI/Y4LVV5

[2] https://archive.ics.uci.edu/dataset/222/bank+marketing
Binarized version available at https://gitlab.tudelft.nl/jgmvanderlinde/dpf [17]

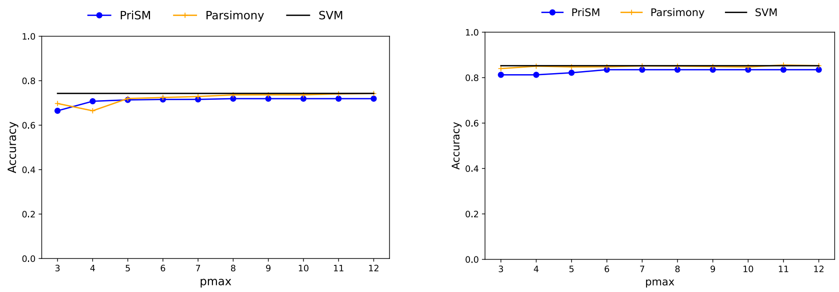
well represents the heterogeneity of real-world data. The dataset collects information about a marketing campaign of a Portuguese bank. It includes 45,211 tuples and is defined over a relational schema including 16 attributes (4 binary, 6 categorical, 6 numerical). The binarized version has been obtained performing a one-hot encoding on all the attributes, after proper binning of numerical attributes. We consider attribute `default`, defined over a binary domain and representing whether the customer has credits in default, as sensitive. The label is attribute `class` defined over a binary domain and indicating whether the customer subscribed a bank term deposit. The frequency distributions of the two values of the sensitive attribute and of the two values of the label are highly unbalanced (more than 98% of occurrences of the non-critical value for attribute `default`, and 88% of 0 for attribute `class`).

5.2 Results

To assess the effectiveness of our PriSM approach, we evaluated: *i)* the quality of classification results when introducing constraints to protect critical values of the sensitive attribute, and *ii)* the ability to reconstruct critical values of the sensitive attribute starting from the predictor attribute used by the classifier. To this purpose, we compared the results obtained using three classifiers:

- SVM: a classical SVM classifier (considering Constraints (2)–(5) in Figure 5), which represents our baseline, to assess the impact of protecting against (direct or indirect) release of sensitive information on classification results;
- Parsimony: a parsimonious SVM classifier that limits the number of predictor attributes used by the classifier to at most *pmax* (considering Constraints (2)–(6) in Figure 5), to assess the impact of the parsimony requirement on classification results;
- PriSM: our privacy-friendly classifier (considering all the constraints in Figure 5).

Accuracy. Limiting the number of predictors and constraining the choice of the same to prevent disclosure of sensitive attributes (and of correlations that might reveal them) is expected to reduce the ability of the classifier to predict the label of data items. Figure 6 compares the accuracy of PriSM and of Parsimony with the baseline accuracy obtained using a traditional SVM (black horizontal line), varying the maximum number *pmax* of predictor attributes used by PriSM and Parsimony from 3 to 12. We did not consider values higher than 12 since for higher values Parsimony produces the same solutions as SVM (parsimony requirement was not binding anymore). Also, Constraint 8 (preventing the classifier from using a set of predictors including sensitive correlations) in PriSM formulation does not permit to select more than 9 predictor attributes. In other words, setting $pmax > 9$ produces the same solution as $pmax = 9$. (Note that such a theoretical limit can be discovered computing a hitting set on the set of minimal sensitive correlations, avoiding unnecessary runs of the training phase.) As expected, the accuracy of PriSM grows with the maximum number of



(a) Synthetic dataset

(b) Bank Marketing dataset

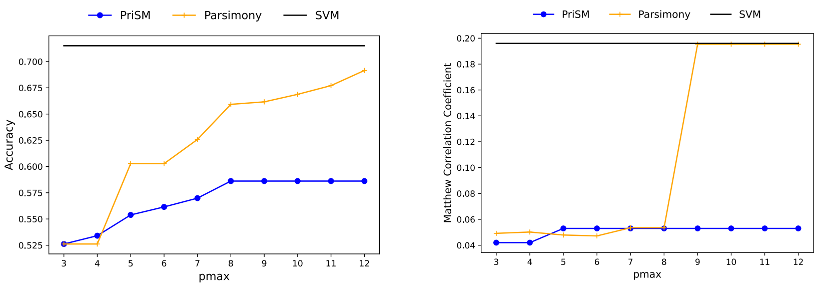
Fig. 6: Accuracy of the classifier varying *pmax*

predictors that the classifier can use. As visible from the figure, protecting the sensitive attribute with PriSM has a limited impact on accuracy, both compared with the results of the SVM baseline and with the ones obtained by Parsimony. Indeed, allowing the classifier to use $pmax \geq 8$ predictors, PriSM loses only 2% accuracy with respect to SVM baseline (Parsimony reaches the same gap with 5 predictors) for both the synthetic and Bank Marketing datasets.

Protection. While noting that PriSM guarantees that no set of attributes with a correlation higher than τ with the sensitive attribute is used by the trained classifier, we empirically analyze the ability of an adversary to predict s based on the predictor attributes used by the classifier produced by PriSM. We then consider a worst case scenario and assume that the adversary trains a classification model for predicting the sensitive attribute values using a dataset having exactly the same distribution of values as the one used by PriSM. In our experiments, this translates in using the same training set for both PriSM and for the adversary model. For simulating the adversary predicting the sensitive attribute, we use ν-SVC (which is a reliable classification model [22]) trained with a nested cross validation, performing hyperparameters tuning in an inner loop.

Figure 7(a) compares the accuracy of PriSM, Parsimony, and SVM in predicting the sensitive attribute values of the synthetic dataset, varying *pmax* from 3 to 12. As visible from the figure, PriSM is much more effective than SVM and Parsimony in protecting the sensitive attribute values. In fact, the accuracy of the adversary classifier remains below 58% with PriSM, while it is about 72% with the SVM. Also with Parsimony the sensitive attribute is gradually more exposed as *pmax* increases, eventually reaching a risk similar to that of SVM. It is interesting to note that, with PriSM, the accuracy of the adversary remains close to the theoretical accuracy lower bound given by the class frequency (52% in our dataset).

Figure 7(b) compares the Matthew's Correlation Coefficient (MCC) of PriSM, Parsimony, and SVM in predicting the sensitive attribute values of the Bank Marketing dataset, varying *pmax* from 3 to 12. We decided to use the MCC for the Bank Marketing dataset since the critical value is very unfrequent (only



(a) Synthetic dataset **(b)** Bank Marketing dataset

Fig. 7: Correlation with the sensitive attribute values varying *pmax*

2% of the tuples). Similarly to what observed on the synthetic dataset, PriSM protects well the sensitive attribute: the MCC score is stable and as low as 0.05 (against 0.20 for SVM). The behavior of Parsimony is instead peculiar: the MCC remains around 0.05 (similar to PriSM) until $pmax$=8, and suddenly worsens at $pmax$=9 reaching the MCC of the SVM. From a more in depth analysis of results, we noticed that this change is due to a strong reduction in the number of false positives, which increases the precision of the adversary model in guessing a positive data item from about 1 case over 50 (which, in practice, approaches a random guess), to about 1 case over 10. This is due to the use by Parsimony of an additional predictor that, combined with the ones already taken, implies the set of predictors to contain a sensitive correlation. Note that in our experiments this happens with $pmax = 9$, but with other training rounds and other cross validation splittings it could happen for lower values of $pmax$.

The experimental results confirm that PriSM maintains high classification accuracy, while protecting the sensitive attribute and its critical values from indirect disclosure through sensitive correlations.

6 Related Work

Several works have addressed the problem of protecting privacy of data in machine learning scenarios (e.g., [20, 21, 28]). Indeed, sensitive data in the training datasets can be exposed to various inference attacks that could violate data privacy (e.g., [7, 11, 13, 15, 23, 29]). To block such attacks, different privacy-preserving machine learning algorithms have been proposed. Each of these solutions operates at a specific step of the machine learning process (i.e., data acquisition, training, and prediction), with the goal of guaranteeing individuals' privacy and preventing data leakage (e.g., [12]).

Privacy-preserving machine learning approaches follow two main strategies: *i)* protect the dataset (e.g., using anonymization approaches [9]) before using it for training the machine learning model; and *ii)* train the machine learning model using a privacy-preserving approach (e.g., a differentially private training [10, 14]). The adoption of anonymization techniques for protecting the training set, while effective in protecting the privacy of the dataset, clearly reduces the accuracy of the machine learning model trained over it. Therefore, recent proposals have focused on the analysis of such an impact on accuracy and of the provided privacy guarantees (e.g., [4, 24]). Other proposals instead addressed the problem of developing utility-aware anonymization techniques that protect data and, at the same time, preserve as much as possible utility for the data analytics working downstream (e.g., [3]). An alternative to anonymization for protecting training datasets is represented by homomorphic encryption that, together with secure multi-party computation, can be used to protect the training dataset while used to train a machine learning model (e.g., [5]). Cryptographic-based approaches, however, typically imply higher computational overhead and loss of precision [12]. The problem of protecting training datasets including confidential information requires particular attention when different parties contribute with their data to the training phase. Indeed, while aiming at collaboration, each data owner also needs to keep their dataset confidential to the other owners (e.g., [19, 26]). PriSM differs from all these proposals since it does not use privacy-preserving techniques neither on the training dataset nor on the machine learning model. The focus is on protecting user's privacy during the prediction phase, ensuring that the machine learning model does not ask (directly or indirectly) any sensitive information as input to provide an accurate prediction.

Related lines of work addressed the problems of overlearning and of fair learning. Overlearning happens when a machine learning model unintentionally learns sensitive attributes (which are not even correlated with the target label) at inference time [25]. Fair learning instead is concerned with producing accurate machine learning models without learning bias like, for instance, in case of unbalance among classes (e.g., [18]). While related, these lines of work are orthogonal to the problem addressed in this paper.

7 Conclusions

We proposed an approach, PriSM, for generating a privacy-friendly classifier that requires neither sensitive information nor information correlated with it for classifying user's data. This goal is reached by first identifying sets of attributes that could (indirectly) reveal the sensitive attribute, and then training the classifier excluding the sensitive attribute as well as other sets of attributes that have been learned as correlated to it. The formulation of the problem as a mixed integer programming problem guarantees protection of the sensitive attribute and misclassification minimization, keeping training times under control. The experiments, performed over both a synthetic and a real-world datasets, confirm that PriSM protects sensitive information also against inference channels due to correlated information, minimizing impact on classification accuracy. The paper leaves space for future works, including the consideration of different families of classifiers (e.g., non linear classifiers) as well as other data analytics tasks.

Acknowledgements. This work was supported in part by the EC under projects GLACIATION (101070141) and EdgeAI (101097300), by the Italian MUR under PRIN project POLAR (2022LA8XBH), and by project SERICS (PE00000014) under the MUR NRRP funded by the EU - NGEU. Project EdgeAI is supported by the Chips Joint Undertaking and its members including top-up funding by Austria, Belgium, France, Greece, Italy, Latvia, Netherlands, and Norway under grant agreement No. 101097300. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union, the Chips Joint Undertaking, or the Italian MUR. Neither the European Union, nor the granting authority, nor Italian MUR can be held responsible for them.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proc. of VLDB. Santiago, Chile (September 1994)
2. Barbato, M., Ceselli, A.: Mathematical programming for simultaneous feature selection and outlier detection under l1 norm. European Journal of Operational Research **316**(3), 1070–1084 (August 2024)
3. Barezzani, S., De Capitani di Vimercati, S., Foresti, S., Ghirimoldi, V., Samarati, P.: TA_DA: Target-aware data anonymization. IEEE TP **2**, 15–26 (2025)
4. Caruccio, L., Desiato, D., Polese, G., Tortora, G., Zannone, N.: A decision-support framework for data anonymization with application to machine learning processes. Information Sciences **613**, 1–32 (October 2022)
5. Chen, C., Wei, L., Xie, J., Shi, Y.: Privacy-preserving machine learning based on cryptography: A survey. ACM TKDD **19**(4) (May 2025)
6. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**, 273–297 (1995)
7. Coscia, P., Ferrari, S., Piuri, V., Salman, A.: Synthetic and (Un)secure: Evaluating generalized membership inference attacks on image data. In: Proc. of SECRYPT. Bilbao, Spain (June 2025)
8. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Livraga, G., Paraboschi, S., Samarati, P.: Fragmentation in presence of data dependencies. IEEE TDSC **11**(6), 510–523 (November/December 2014)
9. De Capitani di Vimercati, S., Foresti, S., Livraga, G., Samarati, P.: k-Anonymity: From theory to applications. Transactions on Data Privacy **16**(1), 25–49 (January 2023)
10. Demelius, L., Kern, R., Trügler, A.: Recent advances of differential privacy in centralized deep learning: A systematic survey. ACM CSUR **57**(6), 1–28 (February 2025)
11. Dick, T., Dwork, C., Kearns, M., Liuc, T., Roth, A., Vietri, G., Wu, Z.: Confidence-ranked reconstruction of census microdata from published statistics. PNAS **120**(8) (2023)
12. El Mestari, S., Lenzini, G., Demirci, H.: Preserving data privacy in machine learning systems. COSE **137** (February 2024)
13. Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., Ristenpart, T.: Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In: Proc. of USENIX. San Diego, CA, USA (August 2014)

14. Jayaraman, B., Evans, D.: Evaluating differentially private machine learning in practice. In: Proc. of USENIX. Santa Clara, CA, USA (August 2019)
15. Jia, J., Gong, N.Z.: AttriGuard: A practical defense against attribute inference attacks via adversarial machine learning. In: Proc. of USENIX. Baltimore, MD, USA (August 2018)
16. Larose, D.T.: Data Mining and Predictive Analytics. Wiley (2015)
17. van der Linden, J., de Weerdt, M., Demirović, E.: Fair and optimal decision trees: A dynamic programming approach. Advances in Neural Information Processing Systems **35**, 38899–38911 (2022)
18. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A.: A survey on bias and fairness in machine learning. ACM CSUR **54**(6), 1–35 (July 2022)
19. Mohassel, P., Zhang, Y.: SecureML: A system for scalable privacy-preserving machine learning. In: Proc. of IEEE S&P. San Jose, CA, USA (May 2017)
20. Rao, B., Zhang, J., Wu, D., Zhu, C., Sun, X., Chen, B.: Privacy inference attack and defense in centralized and federated learning: A comprehensive survey. IEEE TAI **6**(2) (February 2025)
21. Rigaki, M., Garcia, S.: A survey of privacy attacks in machine learning. ACM CSUR **56**(4), 1–34 (April 2023)
22. Scholkopf, B., Smola, A., Williamson, R., Bartlett, P.: New support vector algorithms. Neural Computation **12**(5), 1207–1245 (2000)
23. Shokri, R., Stronati, M., Song, C., Shmatikov, V.: Membership inference attacks against machine learning models. In: Proc. of IEEE S&P. San Jose, CA, USA (May 2017)
24. Slijepčević, D., Henz, M., Klausner, L., Dam, T., Kieseberg, P., Zeppelzauer, M.: k-anonymity in practice: How generalisation and suppression affect machine learning classifiers. COSE **111** (December 2021)
25. Song, C., Shmatikov, V.: Overlearning reveals sensitive attributes. In: Proc. of ICLR. virtual (April/May 2020)
26. Vaidya, J., Yu, H., Jiang, X.: Privacy-preserving SVM classification. Knowledge and Information Systems **14**, 161–178 (2008)
27. Wolsey, L.: Integer Programming. John Wiley & Sons, Ltd (2020)
28. Xue, M., Yuan, C., Wu, H., Zhang, Y., Liu, W.: Machine learning security: Threats, countermeasures, and evaluations. IEEE Access **8**, 74720–74742 (2020)
29. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. In: Proc. of NIPS. Vancouver, Canada (December 2019)

A Minimization of the Misclassification Error

Theorem 1 (Correctness of data correlation discovery). *Given a training dataset r defined over relation schema $R(A, s, l)$, with s a sensitive attribute, a set V_S of sensitive values in the domain of s, the maximum number pmax of predictors used by the classifier, and a correlation threshold τ, the procedure in Figure 4 finds the minimal set $\mathcal{X}$ of correlations for s.*

Proof. To prove that the procedure in Figure 4 computes the set $\mathcal{X}$ of minimal correlations for s, we separately prove the three conditions in Definition 1. Note that we consider a monotone correlation function, that is, the correlation between Y and s is higher than or equal to the correlation between X and s, $\forall X \subseteq Y$.

Condition 1. We first prove that each set Y in $\mathcal{X}$ is a sensitive correlation. Since a set Y is inserted into $\mathcal{X}$ only if the correlation between Y and s is above the given threshold τ (lines 9-11), all the sets Y in $\mathcal{X}$ are such that $Y \rightsquigarrow s$.

Condition 2. We prove that $\mathcal{X}$ captures all the sensitive correlations, either directly or through a dominating correlation. To this purpose, we prove by induction that, at the end of the **repeat-until** loop: *i)* $\mathcal{Y}_c$ contains all the subsets of A of cardinality c that do not represent a sensitive correlation, and *ii)* $\mathcal{X}$ includes all the sensitive correlations with at most c attributes.
For $c = 1$ (base case of the induction), $\mathcal{Y}_1$ contains all the singleton sets of attributes in A that do not represent a sensitive correlation for s. Similarly, $\mathcal{X}$ contains all the singleton sets of attributes in A that represent a sensitive correlation for s. Indeed, $\mathcal{Y}_1$ is initialized to the set of all singleton sets $\{a\}$ with $a \in A$ (line 6). The procedure then checks each set Y in $\mathcal{Y}_1$ and removes it from $\mathcal{Y}_1$ (line 10) inserting it into $\mathcal{X}$ (line 11) if the correlation between Y and s is above threshold τ (line 9).
Let us now assume that the hypothesis holds for $c - 1$ with $c > 1$, that is: *i)* $\mathcal{Y}_{c-1}$ contains all subsets of A of cardinality $c - 1$ that do not represent a sensitive correlation, and *ii)* $\mathcal{X}$ includes all the sensitive correlations with at most $c - 1$ attributes. It is immediate to see that the procedure inserts into $\mathcal{X}$ only sensitive correlations including c attributes. In fact, the sets of attributes in $\mathcal{Y}_c$ include c attributes by construction (line 7) and Y is inserted into $\mathcal{X}$ only if the correlation between Y and s is above threshold τ (line 9). Similarly, at the end of the **repeat-until** loop $\mathcal{Y}_c$ contains only subsets of c attributes that do not represent a sensitive correlation for s, because each set Y in $\mathcal{Y}_c$ representing a sensitive correlation for s is removed from $\mathcal{Y}_c$ (line 10). To prove that $\mathcal{Y}_c$ contains all the subsets of interest (i.e., those subsets of A of cardinality c that do not represent a sensitive correlation), suppose, by contradiction, that there exists a set $Y \subseteq A$ of cardinality c such that all its subsets do not represent sensitive correlations, which does not belong to $\mathcal{Y}_c$ when generated at line 7. That would imply that, at the beginning of the **repeat-until** loop, it has not been possible to find c subsets of Y including $c - 1$ attributes in $\mathcal{Y}_{c-1}$ (i.e., at least one of the subsets of Y of cardinality $c - 1$ does not belong to $\mathcal{Y}_{c-1}$). Since, by the induction hypothesis, $\mathcal{Y}_{c-1}$ contains all the subsets of attributes of cardinality $c-1$ that do not represent sensitive correlations, the missing subset of Y would be a sensitive correlation, therefore also Y would represent a sensitive correlation, thus leading to contradiction. Similarly, to prove that $\mathcal{X}$ includes all the sensitive correlations with at most c attributes, we start from the observation that $\mathcal{X}$ includes all the sensitive correlations with at most $c - 1$ attributes at the beginning of the **repeat-until** loop by hypothesis. Let us assume, by contradiction, the existence of a set Y of cardinality c that represents a sensitive correlation, which is not included in $\mathcal{X}$ at the end of the **repeat-until** loop. This means that Y either has not been inserted into $\mathcal{Y}_c$ at line 7, or it has not been removed from $\mathcal{Y}_c$ at line 10. Since Y represents a sensitive correlation, it cannot be maintained in $\mathcal{Y}_c$ since the condition at line 9 would be satisfied by Y. If Y is not inserted into $\mathcal{Y}_c$ at line 7, it means that it does not have c subsets of

$c-1$ attributes each in $\mathcal{Y}_{c-1}$, that is, that do no represent a sensitive correlation. Since a set of c attributes has exactly c subsets of $c-1$ attributes, if not all these subsets belong to $\mathcal{Y}_{c-1}$, there is at least a subset of Y representing a sensitive correlation that, by induction hypothesis, already belongs to $\mathcal{X}$. Therefore, Y is already represented in $\mathcal{X}$.

Since the invariant holds for each value of c, it holds also for $c = pmax$ (and, in the worst case, for $c = |A|$), thus proving that $\mathcal{X}$ captures all the sensitive correlations.

Condition 3. We prove that $\mathcal{X}$ does not include any sensitive correlation that is a superset of another sensitive correlation in $\mathcal{X}$. The satisfaction of this condition follows by construction of sets $\mathcal{Y}_c$. As illustrated above, for Y to be included in $\mathcal{X}$, it must be generated as a candidate sensitive correlation as the union of c sets of attributes of cardinality $c-1$ that do not represent sensitive correlations (line 7). Therefore, the condition holds. □

Theorem 2 (Correctness of PriSM). *Given a training dataset r defined over relation schema $R(A, s, l)$, with s the sensitive attribute and l the label attribute, and the minimal set $\mathcal{X}$ of sensitive correlations for s,* PriSM *computes a privacy-friendly classifier that minimizes misclassification (i.e., solves Problem 1).*

Proof. Since any classifier computed as a solution to the Mixed Integer Programming formulation of PriSM training problem satisfies all the constraints in Figure 5, the classifier is privacy-friendly (Definition 2). Indeed, Constraint 7 excludes the sensitive attribute s from the set of predictors, and Constraint 8 excludes from the set of predictors at least one attribute for each sensitive correlation $X\in\mathcal{X}$. In fact, Constraint 7 is satisfied only if $\mathbf{p}[s]$ is 0, and Constraint 8 is satisfied only if $\mathbf{p}[a]$ is 0 for at least one attribute a in X, for each $X\in\mathcal{X}$. Thanks to Constraint 5, if $\mathbf{p}[a]=0$ then $\mathbf{w}[a]=0$. Therefore, any solution to the Mixed Integer Programming problem in Figure 5 is privacy-friendly (Definition 2).

Since the Mixed Integer Programming formulation of PriSM in Figure 5 defines a binary variable $\mathbf{p}[a]$ for each candidate predictor attribute in $R \setminus \{l\}$, it implicitly encodes all the (combinatorially many) possible choices of subsets of $R \setminus \{l\}$ as predictors. Constraint 6 limits to at most $pmax$ the number of attributes for which $\mathbf{p}[s]=1$, and then the number of predictors. Solving the problem in Figure 5 is then equivalent to (implicitly) explore all the possible choices of predictors as subsets of $R \setminus \{l\}$ of cardinality at most $pmax$.

Relaxing integrality conditions on $\mathbf{p}$ variables, we obtain a continuous optimization problem. Such a residual optimization problem has a quadratic convex objective function (it is the sum of a linear function and a squared norm-2 term) and linear constraints. It is therefore a convex optimization problem, which can be solved to proven global optimality by means of many effective algorithms. Branch-and-bound, branch-and-cut [27] or even more effective algorithms can therefore be used to solve the problem in Figure 5 to proven global optimality, in terms of both choice of predictors and final hyperplane. Since the objective function of the formulation of PriSM in Figure 5 is the classical objective function of the Mixed Integer Programming formulation of the SVM problem, the solutions to the problem in Figure 5 minimize misclassification. □