Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Cleaning the Null Space: A Privacy Mechanism for Predictors

Ke Xu,* **Tongyi Cao,†** **Swair Shah,***
Crystal Maung,* **Haim Schweitzer***
{ke.xu5,swair,hschweitzer}@utdallas.edu,
tcao@umass.edu,
Crystal.Maung@gmail.com

Abstract

In standard machine learning and regression setting feature values are used to predict some desired information. The privacy challenge considered here is to prevent an adversary from using available feature values to predict confidential information that one wishes to keep secret. We show that this can sometimes be achieved with almost no effect on the quality of predicting desired information. We describe two algorithms aimed at providing such privacy when the predictors have a linear operator in the first stage. The desired effect can be achieved by zeroing out feature components in the approximate null space of the linear operator.

Introduction

Consider a company that develops technology for predicting desired information from raw data. As a trivial example the raw data may be the height, weight, and sex of a child, with the desired information being the child's age. The company may obtain the data from a client, use it to predict the desired information, and send that information back to the client.

The privacy concern we address is the potential inappropriate use of the client data to predict confidential information that the client does not wish to expose. In our trivial example such information may be the child weight (which is part of the data), or whether or not the child is obese.

It appears that the current approach in similar situations is to encrypt the raw data sent to the company so that it requires a secret password to be accessed. This, however, does not provide significantly improved privacy. Someone who works at the company and knows the password may still access the data, and an adversary may still gain access to both the encrypted data and the password.

Another possible solution is to provide the client with the software that computes the desired information, so that the raw data need not be sent to the company. This solution is sometimes inappropriate because of the following two reasons. First, the software may be too big and require special hardware. Secondly, giving the program away may allow the client to use it in cases not agreed upon by the company. Similar arguments are made in (Tramèr et al. 2016).

*Department of Computer Science, Univ. of Texas at Dallas.
†CICS, UMass Amherst.

In this paper we propose a privacy mechanism to minimize the exposure of confidential information that the client may wish to keep private even from the company. We show how to "clean" the data before sending it to the company in such a way that allows its usage for predicting the desired information. At the same time the accuracy of predicting confidential information from the cleaned data is significantly reduced.

The main idea behind our approach can be easily understood in terms of feature selection. Clearly, it makes no sense to provide the company data that is not needed for predicting the desired information. Unfortunately, if accurate prediction of desired labels is necessary, the removal of an entire feature may not be appropriate. Even "mostly irrelevant" features may contribute "something" to the prediction. Instead, we show how to remove unnecessary feature components.

Our idea is to use knowledge about the predictor of desired information to help hide confidential information. In particular, if the predictor has a linear operator in the first stage then there is a transformation of the data that reveals feature components not needed for the prediction. Specifically, combinations of features that lie in the operator null space do not affect the prediction and need not be provided. Removing this information is what we call *cleaning*.

Problem statement

The problem we consider involves three entities: the ***cleaner***, the ***ally***, and the ***adversary***. In the example discussed earlier the cleaner is the client and the ally is the company. The cleaner has a single feature vector denoted by x. In addition to x the cleaner has knowledge of the linear operator of the predictor used by the ally. The cleaner cleans x and produces $\tilde{x}$. We write the computation performed by the cleaner as:

$$\tilde{x} = \text{clean}(x) \tag{1}$$

The ally has a predictor f_d that can predict desired information, denoted by y_d, from the uncleaned data x. This can be expressed as: $y_d = f_d(x)$. The ally receives the cleaned feature vector $\tilde{x}$ from the cleaner without knowledge of whether or not it was cleaned, or how it was cleaned. The computation performed by the ally is always the same, applying the

predictor to the data:

$$\tilde{y}_d = f_d(\tilde{x}) \tag{2}$$

The adversary attempts to predict confidential information denoted by y_c. It does not know x, but it knows everything else. In particular it knows the cleaned feature vector $\tilde{x}$, the method clean() which was used by the cleaner, and the method $f_d()$ which was used by the ally. In addition, the adversary may have training data in the form of (x^i, y_c^i), relating uncleaned feature vectors to confidential information. We assume that the training data is good enough to infer an accurate predictor of the form: $y_c = f_c(x)$. In order to discover the confidential information the adversary calculates a predictor $\tilde{f}_c$ and produces the following approximation:

$$\tilde{y}_c = \tilde{f}_c(\tilde{x}) \tag{3}$$

Evaluation criteria

We wish to obtain algorithms for "clean()" such that the vector $\tilde{x}$ produced by the cleaner satisfies:

$$\begin{aligned} f_d(\tilde{x}) = \tilde{y}_d \approx y_d, \quad & e_{\text{utility}} = |\tilde{y}_d - y_d|^2 \\ \tilde{f}_c(\tilde{x}) = \tilde{y}_c \not\approx y_c, \quad & e_{\text{privacy}} = |\tilde{y}_c - y_c|^2 \end{aligned} \tag{4}$$

We say that the cleaning method has *high utility* if e_{utility} is guaranteed to be small, and that it has *high privacy* if e_{privacy} is guaranteed to be large. We describe algorithms that provide a guarantee on utility but not for the worst case privacy. Specifically, for any given ϵ and x the algorithms guarantee cleaning that satisfies:

$$e_{\text{utility}} = |\tilde{y}_d - y_d|^2 \leq \epsilon \tag{5}$$

In our model, the adversary knows everything except for the feature vector x. It is impossible to guarantee privacy in the worst case because of the following argument. If $y_c = y_d$ then the adversary can use the predictor f_d to predict the confidential information. Therefore, in this worst case high utility would imply low privacy. Instead of considering the worst case we attempt to maximize e_{privacy} while still satisfying the constraint on e_{utility}. We present two algorithms. The first maximizes the "expected privacy error" that will be defined later. The second attempts to learn the predictor used by the adversary and maximize the privacy protection against that particular predictor.

Our results

Our cleaning algorithms require that the predictor $f_d()$ starts with a linear operator. This means that it can be written as: $f_d(x) = f_1(A_d^T x)$, where A_d is a matrix. (Extensions to the nonlinear case are nontrivial.) Here are several examples of commonly used predictors that can be expressed in this way.

1. Linear regression (e.g., (Miller 2002; Hastie, Tibshirani, and Friedman 2009)).
2. Many approaches to multi-label classification start with linear regression. The regression results are then followed by thresholding or other classification schemes, converting the real valued data to discrete values. See, e.g. (Tsoumakas et al. 2011; Zhang and Zhou 2014).
3. Multi-layer neural nets start with a linear operator applied to the input data. (See, e.g., (Hastie, Tibshirani, and Friedman 2009)).
4. Algorithms that use linear dimensionality reduction such as PCA. See, e.g., (Jolliffe 1986).

We describe two cleaning algorithms that work intuitively as follows. The algorithms take as input the feature vector x and the matrix A_d (used by the ally in the initial linear step). The cleaning of x is achieved by subtracting from it projections on the approximate null space of A_d^T. These projections are irrelevant to the prediction of the desired information. Both algorithms remove components in the exact null space. They behave differently in identifying the approximate null space.

A toy example

To illustrate the main ideas behind our approach consider the following toy example:

(x_1, x_2)	cleaned (x_1, x_2)	y_d	y_c
(3,1)	(1,-1)	2	5
(4,2)	(1,-1)	2	8
(5,1)	(2,-2)	4	7
(6,5)	(0.5,-0.5)	1	16

There are two features x_1, x_2, one desired label y_d, and one confidential label y_c. Both y_d and y_c can be calculated exactly from x_1, x_2:

$$y_d = x_1 - x_2, \quad y_c = x_1 + 2x_2$$

Clearly, the desired information y_d can be predicted exactly from x_1, x_2 with zero error. However, y_c is exposed. Here the prediction model of y_d is $A_d^T = (1, -1)$, with the vector $(1, 1)$ in its null space. Thus, the vector (x_1, x_2) can be cleaned by zeroing out projections on the direction $(1, 1)$. This amounts to subtracting the mean of x_1 and x_2 from each coordinate.

After cleaning, the same linear model can still be used to predict y_d from the cleaned features. On the other hand there is no predictor for computing y_c exactly from the cleaned features, since two different values of y_c must be inferred from two identical cleaned feature vectors (lines 1,2).

Relation to previous work

Adapting machine learning terminology of a learning and a testing phase, our goal is protecting the privacy of information during the testing phase. It is different from studies concerned with the privacy of training data, such as differential privacy e.g., (Dwork and Roth 2014; Sarwate and Chaudhuri 2013), where noise is added to blur the distinction between individual items in the training data.

There are several studies that investigate feature selection as a tool for obtaining privacy of training data. See, e.g., (Pattuk et al. 2015; Jafer, Matwin, and Sokolova 2015; Banerjee and Chakravarty 2011). The idea is not to release the entire information in the data, but only selected features.

Unlike these studies we consider the privacy of information that can be extracted from a single feature vector.

1. If $H_j = 0$ set $\alpha_j = 1$.
2. Sort the pairs (G_j, H_j) in decreasing order of G_j/H_j. Let (j) be the location of (G_j, H_j) in this order.
3. Compute the index t such that:
$$\sum_{(j)=1}^{t} H_{(j)} < \epsilon, \quad \sum_{j=1}^{t+1} H_{(j)} \geq \epsilon$$
4. Set $r = (\epsilon - \sum_{(j)=1}^{t} H_{(j)})/H_{(t+1)}$
5. Set $\alpha_{(j)} = \begin{cases} 1 & 1 \leq (j) \leq t \\ r & (j) = t+1 \\ 0 & \text{otherwise} \end{cases}$

Figure 1: Solving the Fractional Knapsack

Clearly, to increase privacy we can add noise to the feature vector or use fewer features. However, with no distinction between desired and and confidential information the increase in privacy would imply a decrease in accuracy.

Recent studies (Enev et al. 2012; Hamm 2015; Whitehill and Movellan 2012) consider a setting similar to ours. They also make the distinction between desired and confidential information. In (Hamm 2015), the ally (data aggregator) and the user (data contributor) participate in the data filtering. The goal is to prevent the adversary from predicting the private information of user while maintaining the utility of the data. The framework proposed in (Enev et al. 2012) transforms the data in a way that the covariance between the data and the desired information is increased, while the covariance between the data and confidential information is decreased. The most important difference between our approach and these approaches is in the treatment of the predictors. They make different (or no) assumptions about the predictors, while we assume knowledge of these predictors, which yields different algorithms.

Optimization tool

The algorithms we propose require an optimization tools that is detailed in this section. The problem to be optimized is a variation on the classical fractional Knapsack problem, originally discussed by Dantzig in (Dantzig 1957). See also (Goodrich and Tamassia 2002).

The Fractional Knapsack
1. Input: $G_1, \ldots, G_n$, nonnegative $H_1, \ldots, H_n$, ϵ.
2. Output: $\alpha_1, \ldots, \alpha_n$ that solve the following problem:
3. Maximize $\sum_j \alpha_j G_j$.
4. Subject to: $\sum_j \alpha_j H_j \leq \epsilon$,
5. $0 \leq \alpha_j \leq 1$.

The standard algorithm for the optimal solution to this problem is shown in Fig. 1. We need the following variation of the fractional knapsack.

Augmented objective function and equality constraint. Here both the objective function in line 3 and the equality constraint in line 4 are replaced by:

3. Maximize $\sum_j \alpha_j^2 G_j$.

Input: x: the feature vector to be cleaned,
A_d: the known predictor,
ϵ: the desired value of e_{utility}.
Output: the cleaned feature vector $\tilde{x}$.
1. Compute eigenvectors/eigenvalues of $B_d = A_d A_d^T$.
2. For each eigenvector/eigenvalue pair (v_j, λ_j) compute: $a_j = v_j^T x$, $\delta_j = \lambda_j a_j^2$.
3. Solve the Augmented Fractional Knapsack problem $G_j = 1$, $H_j = \delta_j$, ϵ, to determine α_j, $j = 1, \ldots, n$.
4. Output $\tilde{x} = x - \sum_{j=1}^{n} \alpha_j a_j v_j$.

Figure 2: Algorithm 1 for cleaning a feature vector x.

4. Subject to: $\sum_j \alpha_j^2 H_j \leq \epsilon$.

The only change needed in the algorithm of Fig.1 is in line 4.

4. Set $r = \sqrt{(\epsilon - \sum_{(j)=1}^{t} H_{(j)})/H_{(t+1)}}$

Cleaning

In this section we describe the algorithms for cleaning the feature components in the approximate null space of the linear predictor. As in the previous section we denote the feature vector by x, and the cleaned feature vector by $\tilde{x}$. To simplify the discussion and the notation we assume that the predictor f_d is linear, so that $f_d(x) = A_d^T x$.

Algorithm 1

The first algorithm is shown in Fig. 2. It identifies t eigenvectors and a fraction α_{t+1} of another eigenvector that are in the approximate null space of A_d. Zeroing out the projection of the feature vector x on these eigenvectors produces the desired cleaned feature vector.

Theorem 1: Let $\tilde{x}$ be the result of cleaning x by Algorithm 1. Set y_d=$A_d^T x$, $\tilde{y}_d$=$A_d^T \tilde{x}$, and $e_{\text{utility}} = |y_d - \tilde{y}_d|^2$. Then $e_{\text{utility}} \leq \epsilon$.

Proof: With the notation of Algorithm 1:

$$y_d - \tilde{y}_d = A_d^T x - A_d^T \tilde{x} = A_d^T (x - \tilde{x}) = A_d^T \sum_{j=1}^{n} \alpha_j a_j v_j$$

Therefore:

$$e_{\text{utility}} = |y_d - \tilde{y}_d|^2 = (\sum_{i=1}^{n} \alpha_i a_i v_i^T) B_d (\sum_{j=1}^{n} \alpha_j a_j v_j)$$

$$= \sum_{i=1}^{n} \lambda_i \alpha_i^2 a_i^2 = \sum_{i=1}^{t} \lambda_i a_i^2 + \lambda_{t+1} a_{t+1}^2 r^2 \leq \epsilon \quad \blacksquare$$

The last inequality follows from the promise of the Fractional Knapsack.

We proceed to analyze the privacy properties of Algorithm 1. Ideally, an algorithm should be designed with the following criterion:

$$\text{maximize} \quad \inf e_{\text{privacy}}$$

where the infimum is over all possible algorithms. While this cannot be shown in our model, we can show that Algorithm 1 maximizes the following related criterion:

$$\text{maximize} \quad \mathsf{Exp}\{e_{\text{privacy}}\} \tag{6}$$

where the expectation is over the probability distribution defined below. Suppose the adversary uses the *linear* model A_c to predict the confidential labels from x. Let $\mathbb{A}_{\mathcal{X}}$ be the set of all pairs that can be obtained from the pair (A_c, x) by rotation. Specifically, the pair (A_1, x_1) belongs to $\mathbb{A}_{\mathcal{X}}$ if there is an orthogonal matrix Q_1 such that $A_1 = Q_1 A_c$ and $x_1 = Q_1 x$. The probability distribution in (6) is the uniform distribution over the elements of $\mathbb{A}_{\mathcal{X}}$. (Multiplication by an orthogonal matrix can be viewed as a change of the coordinate system.)

Lemma 1: Let $v_1, \ldots, v_n$ be an orthogonal basis of $\mathbb{R}^n$, and let (A, x) be a random variable in $\mathbb{A}_{\mathcal{X}}$. Then:

$$\mathsf{Exp}\left\{x^T v_i v_i^T A A^T v_j v_j^T x\right\} \text{ is independent of } i, j,$$

where the expectation is with respect to the uniform distribution defined over the elements of $\mathbb{A}_{\mathcal{X}}$.

Proof: For any orthogonal matrix Q we have:

$$\begin{aligned}
&\mathsf{Exp}\left\{x^T v_i v_i^T A A^T v_j v_j^T x\right\} \\
&= \int x^T v_i v_i^T A A^T v_j v_j^T x \; \mathsf{prob}((A,x)) d((A,x)) \\
&= \int v_j^T x x^T v_i v_i^T A A^T v_j \; \mathsf{prob}((A,x)) d((A,x)) \\
&= \int v_j^T Q x x^T Q^T v_i v_i^T Q A A^T Q^T v_j \; \mathsf{prob}((QA,Qx)) \\
&\qquad\qquad d((QA,Qx)) \\
&= \int v_j^T Q x x^T Q^T v_i v_i^T Q A A^T Q^T v_j \; \mathsf{prob}((A,x)) d((A,x)) \\
&= \mathsf{Exp}\left\{v_j^T Q x x^T Q^T v_i v_i^T Q A A^T Q^T v_j\right\} \\
&= \mathsf{Exp}\left\{x^T Q^T v_i v_i^T Q A A^T Q^T v_j v_j^T Q x\right\} \qquad (7)
\end{aligned}$$

(The integrals are computed over $\mathbb{A}_{\mathcal{X}}$.) To prove the lemma for the case where $i{=}j$ we need to show that $\mathsf{Exp}\left\{x^T v_i v_i^T A A^T v_i v_i^T x\right\} = \mathsf{Exp}\left\{x^T v_k v_k^T A A^T v_k v_k^T x\right\}$ for $1 \le k \le n$. This follows immediately by applying Equation (7) with any orthonormal matrix Q satisfying $v_k = Q^T v_i$.

For the case where $i{\neq}j$ we need to show that $\mathsf{Exp}\left\{x^T v_i v_i^T A A^T v_j v_j^T x\right\} = \mathsf{Exp}\left\{x^T v_k v_k^T A A^T v_l v_l^T x\right\}$ for $1 \le k, l \le n$, $k \neq l$. This follows immediately by applying Equation (7) with any orthonormal matrix Q satisfying $v_k = Q^T v_i$, $v_l = Q^T v_j$. Such orthogonal matrix Q always exists since there is always a "change-of-basis" orthogonal matrix that maps one basis of $\mathbb{R}^n$ to another. See, e.g., (Golub and Van-Loan 2013). ■

Input: x: the feature vector to be cleaned,
A_d: the known predictor,
ϵ: the desired value of e_{utility},
training data: (x^i, y_c^i).

Output: the cleaned feature vector $\tilde{x}$.

1. Use the training data to compute the predictor A_c so that $y_c^i \approx A_c^T x^i$.
2. For $B_d = A_d A_d^T$, $B_c = A_c A_c^T$, solve the generalized eigenvalue problem: $B_d v = \gamma B_c v$. The result is $(v_1, \gamma_1), \ldots, (v_n, \gamma_n)$.
3. for $j = 1, \ldots, n$: $a_j = v_j^T x$, $\lambda_j^d{=}v_j^T B_d v_j$, $\delta_j^d{=}\lambda_j^d a_j^2$, $\lambda_j^c{=}v_j^T B_c v_j$, $\delta_j^c{=}\lambda_j^c a_j^2$.
4. Solve the Augmented Fractional Knapsack for $G_j = \delta_j^c$, $H_j = \delta_j^d$, ϵ, to determine α_j, $j = 1, \ldots, n$.
5. Output $\tilde{x} = x - \sum_{j=1}^n \alpha_j a_j v_j$.

Figure 3: Algorithm 2 for cleaning a feature vector x.

Theorem 2: Suppose the predictor of confidential information from x is linear, represented by the matrix A_c. Let $\mathbb{A}_{\mathcal{X}}$ be the set of all rotations of the pair (A_c, x), and consider a uniform probability distribution over the elements of $\mathbb{A}_{\mathcal{X}}$. Then the cleaned vector produced by Algorithm 1 maximizes the expected privacy error.

Proof: As in the proof of Theorem 1 we have:

$$\begin{aligned}
e_{\text{privacy}} &= |y_c - \tilde{y}_c|^2 \\
&= x^T (\sum_{i=1}^n \alpha_i v_i v_i^T) A_c A_c^T (\sum_{j=1}^n \alpha_j v_j v_j^T) x \\
&= \sum_{ij} \alpha_i \alpha_j x^T v_i v_i^T A_c A_c^T v_j v_j^T x
\end{aligned}$$

Taking expectations gives:

$$\mathsf{Exp}\{e_{\text{privacy}}\} = b \sum_{ij} \alpha_i \alpha_j = b (\sum_j \alpha_j)^2$$

where $b = \mathsf{Exp}\left\{x^T v_i v_i^T A A^T v_j v_j^T x\right\}$ which according to Lemma 1 is independent of i, j. Therefore, maximizing $\mathsf{Exp}\{e_{\text{privacy}}\}$ in terms of the α_j requires solving the following optimization problem:

$$\text{Maximize} \quad \sum_j \alpha_j \quad \text{s.t.} \;\; 0 \le \alpha_j \le 1, \;\; \sum_j \alpha_j^2 \delta_j \le \epsilon$$

And this can be solved as the Augmented Fractional Knapsack. ■

Algorithm 2

The second algorithm is shown in Fig. 3. The cleaner uses its own training data to estimate a linear predictor A_c for the confidential labels. It then proceeds similarly to Algorithm 1, but the approximate null space is computed in terms of generalized eigenvectors and eigenvalues.

Theorem 3: Let $\tilde{x}$ be the result of cleaning x by Algorithm 2. Set y_d=$A_d^T x$, $\tilde{y}_d$=$A_d^T \tilde{x}$, and $e_{\text{utility}} = |y_d - \tilde{y}_d|^2$. Then $e_{\text{utility}} \leq \epsilon$.

Proof: Identical to the proof of Theorem 1. ■
We proceed to analyze the privacy properties of Algorithm 2.

Theorem 4: If the Adversary uses the linear predictor A_c as computed by the cleaner then the cleaned vector produced by Algorithm 1 maximizes the privacy error.

Proof: The generalized eigenvalues γ_j satisfy:

$$\gamma_j = \lambda_j^d / \lambda_j^c, \quad \text{where } \lambda_j^d = v_j^T B_d v_j,\ \lambda_j^c = v_j^T B_c v_j$$

Following the same derivation as in Theorem 1 we have:

$$e_{\text{utility}} = \sum_{j=1}^{n} \alpha_j^2 \delta_j^d$$

$$e_{\text{privacy}} = \sum_{j=1}^{n} \alpha_j^2 \delta_j^c$$

Therefore, the optimization problem that needs to be solved to maximize the privacy error is:

$$\text{Maximize } \sum_{j=1}^{n} \alpha_j^2 \delta_j^c$$

$$\text{s.t. } 0 \leq \alpha_j \leq 1, \quad \sum_j \alpha_j^2 \delta_j^d \leq \epsilon$$

And this problem is solved by Algorithm 2 using the Augmented Fractional Knapsack. ■

Experimental results

We evaluated the proposed algorithms on benchmark datasets from Mulan repository (Tsoumakas et al. 2011) shown in Table 1. Experimental results are averaged over 10 runs. In each run 90% of the dataset is chosen randomly to compute the model of the ally. The rest is used as testing data, where each individual sample is evaluated independently of the other samples. We ran three experiments with each dataset, varying the number of desired and confidential labels:

1) one desired label and all the rest are confidential labels.
2) randomly select half of the labels as desired and the other half as confidential.
3) one confidential and all the rest are desired labels.

In the tables N_d and N_c denote the number of desired and confidential labels respectively. The results are shown in Fig 4. Observe that in almost all cases e_{utility} is exactly ϵ and e_{privacy} is much bigger than e_{utility}. The special case happens on dataset CAL500 when N_d is 1 and N_c is 173.

To provide further insight into the performance of the cleaning mechanism we count the test cases for which the cleaning achieves "complete privacy". The criterion for ***complete privacy*** is defined as follows.

name	instances(m)	features(n)	labels(N)
scene	2407	294	6
wq	1060	16	14
oes97	334	263	16
CAL500	502	68	174

Table 1: Dataset description.

> If the error of predicting y_c using $\tilde{x}$ is greater than the error of predicting y_c from the mean feature vector, then ***complete privacy*** has been achieved.

In Fig 4, Complete Privacy denotes the percentage of test cases for which the cleaner achieves *complete privacy*. We observe that in most of the cases the adversary would be no more advantageous using the cleaned feature vector $\tilde{x}$ than using the mean feature vector.

In some cases, Algorithm 1 and Algorithm 2 have the same performance (e.g. dataset scene and oes97). In other cases (e.g. dataset CAL500 and wq), the two algorithms perform differently. It is because both the algorithms will clean feature vector x with the eigenvectors/generalized eigenvectors in the nullspace first. These vectors are identical to the two algorithms. After such cleaning e_{utility} maintains 0. To achieve the desired value of e_{utility}, both the algorithms will keep cleaning x with other vectors/generalized eigenvectors. This is the procedure where Algorithm 1 and Algorithm 2 diverges. Algorithm 2 selects vectors obtained by taking into consideration the predictor model A_c for confidential information, while Algorithm 1 selects vectors obtained without considering such information.

It is possible that the adversary uses a model other than A_c which may be better suited for predicting confidential labels. We discuss such a scenario next.

A Proposed Attack

We assume the adversary knows the cleaning algorithms. Therefore, the adversary can acquire a dataset and obtain cleaned features. With the information of confidential labels and features after cleaning, a new model can be inferred for predicting y_c. Given a test example, the adversary would apply the new model to make prediction. The procedure is shown in Fig 5.

The experiments with the proposed attack show that e_{utility} is still exactly ϵ for both algorithms in almost all the cases. Due to the limitation of table size, here e_{utility} is not shown. Other results are shown in Figure 6. In some cases the rate of complete privacy decreases to 40%. That shows the cleaned feature vector provides no more confidential information than the mean feature vector for 40% of the test samples. However, e_{privacy} is still considerably higher than e_{utility}. It is to be noted that for some datasets e_{privacy} becomes extremely high (e.g. oes97) and for some datasets e_{privacy} is relatively small. This is dependent on the dataset as well as the desired and confidential labels. A higher privacy (at the expense of utility) can be obtained by increasing the value of ϵ.

Dataset	N_d	N_c	Algorithm 1			Algorithm 2		
			e_{utility}	e_{privacy}	Complete Privacy	e_{utility}	e_{privacy}	Complete Privacy
scene	1	5	0.01	0.796	81.8%	0.01	0.796	81.8%
	3	3	0.01	1.358	80.6%	0.01	1.296	80.7%
	5	1	0.01	4.259	82.3%	0.01	3.659	82.3%
CAL500	1	173	0.009	0.845	85.7%	0.01	5.3E+10	100.0%
	87	87	0.01	0.156	63.5%	0.01	1.741	79.9%
	173	1	0.01	0.034	46.0%	0.01	2.188	59.6%
wq	1	13	0.01	2.654	52.6%	0.01	2.654	52.6%
	7	7	0.01	2.172	58.8%	0.01	1.956	57.8%
	13	1	0.01	2.265	74.8%	0.01	1.403	45.5%
oes97	1	15	0.01	5.11E+7	65.4%	0.01	5.11E+7	65.4%
	8	8	0.01	4.23E+7	63.0%	0.01	4.23E+7	63.0%
	15	1	0.01	6.02E+7	47.0%	0.01	6.02E+7	47.0%

Figure 4: Results with no attack, $\epsilon = 0.01$, Almost all the cases e_{utility} is exactly ϵ, and e_{privacy} is much bigger than e_{utility}. Complete Privacy denotes the percentage of test samples which achieve *complete privacy*.

Algorithm 3

Input: x^t: a test example,
training data: (x^i, y_c^i).

Output: Attempted prediction of y_c^t from x^t.

1. Repeat for each feature vector x^i:
Apply the cleaning algorithm to get a cleaned feature vector $\tilde{x}^i$.

2. Use the cleaned training data to build a prediction model.

3. Use the model created in step 2 to predict y_c^t from x^t.

Figure 5: A proposed attack on the cleaning algorithms.

N_d	N_c	Algorithm 1		Algorithm 2	
		e_{privacy}	CP	e_{privacy}	CP
scene dataset (n = 294)					
1	5	0.289	51.0%	0.289	51.0%
3	3	0.235	44.7%	0.233	44.5%
5	1	0.047	41.0%	0.043	41.8%
CAL500 dataset (n = 68)					
1	173	0.230	48.6%	0.005	58.9%
87	87	0.081	56.9%	0.008	55.9%
173	1	0.068	44.0%	0.029	44.4%
wq dataset (n = 16)					
1	13	0.308	48.9%	0.308	48.9%
7	7	0.132	39.2%	0.164	38.7%
13	1	0.092	31.9%	0.086	30.9%
oes97 dataset (n = 263)					
1	15	4.67E+6	39.4%	4.67E+6	39.4%
8	8	7.48E+6	43.6%	7.48E+6	43.6%
15	1	1.16E+7	18.2%	1.16E+7	18.2%

Figure 6: Results with the proposed attack. $\epsilon = 0.01$, CP is the percentage of test cases that achieve *complete privacy*. Still e_{privacy} is bigger than e_{utility} in most of the cases.

Comparison with a Differential Privacy Mechanism

The goal of differential privacy is protecting the individual entry of a database while allowing accurate statistical analysis of the entire database. A standard way to achieve differential privacy is the Laplace mechanism (Dwork and Roth 2014). Carefully adding noise drawn from a Laplace distribution to the features would achieve differential privacy but at the cost of utility. On the other hand if one adds noise to the feature vector while attempting to keep e_{utility} small, then e_{privacy} would also remain very small, thus privacy is sacrificed.

In the experiments, we adjust the Laplace noise parameter *mean* and *scale* to produce the same e_{utility} as Algorithm 1. The results are shown in Fig. 7. Observe that even with the proposed attack our approach achieves better *complete privacy* compared to the Laplace mechanism. Thus for the same value of utility our approach provides much better privacy.

Concluding remarks

The problem considered in this paper is protecting the privacy of predictors. It is different from studies concerned with the privacy of the training data, such as differential privacy. Recent studies about balancing utility and privacy propose similar models with different (or no) assumptions about the predictors. By contrast, we assume partial knowledge of these predictors, which may provide sharper results.

The proposed algorithms can be applied to models that start with a linear operator. We observe that projections on the null space of the predictors do not change the prediction value. These projections give out information that is unnecessary for the prediction. Therefore, cleaning the data by zeroing out projections on the null-space will not affect the prediction accuracy but will increase the privacy. The more distortion in feature vectors, the higher privacy and the lower utility. The trade-off between utility and privacy can be controlled by adjusting the ϵ parameter.

Dataset	e_{utility}	Laplace Mechanism		Alg 1 with no attack		Alg 1 with attack	
		e_{privacy}	Complete Privacy	e_{privacy}	Complete Privacy	e_{privacy}	Complete Privacy
scene	0.01	0.009	31.5%	0.796	81.8%	0.289	51.0%
	0.02	0.019	31.4%	0.772	81.8%	0.289	50.3%
	0.03	0.027	31.3%	0.770	81.8%	0.289	51.1%
oes97	0.01	0.015	24.6%	4.23E+7	63.0%	7.48E+6	43.6%
	0.10	0.162	24.6%	4.23E+7	63.0%	7.48E+6	43.6%
	1.00	1.698	24.6%	4.23E+7	63.0%	7.48E+6	43.5%

Figure 7: Comparison with the Laplace Mechanism. For dataset scene, N_d is 1, N_c is 5. For dataset oes97, N_d is 8, N_c is 8. Our approach provides much better privacy for the same value of utility.

References

Banerjee, M., and Chakravarty, S. 2011. Privacy preserving feature selection for distributed data using virtual dimension. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, CIKM '11, 2281–2284. New York, NY, USA: ACM.

Dantzig, G. B. 1957. Discrete-variable extremum problems. *Operations Research* 5:266–277.

Dwork, C., and Roth, A. 2014. The algorithmic foundations of differential privacy. *Foundions and Trends in Theoretical Computer Science* 9(3 & 4):211–407.

Enev, M.; Jung, J.; Bo, L.; Ren, X.; and Kohno, T. 2012. Sensorsift: balancing sensor data privacy and utility in automated face understanding. In *Proceedings of the 28th Annual Computer Security Applications Conference*, 149–158. ACM.

Golub, G. H., and Van-Loan, C. F. 2013. *Matrix Computations*. Johns Hopkins University Press, fourth edition.

Goodrich, M. T., and Tamassia, R. 2002. *Algorithm Design: Foundations, Analysis, and Internet Examples*. John Wiley & Sons.

Hamm, J. 2015. Preserving privacy of continuous high-dimensional data with minimax filters. In *AISTATS*.

Hastie, T.; Tibshirani, R.; and Friedman, J. 2009. *The Elements of Statistical Learning*. Springer, second edition.

Jafer, Y.; Matwin, S.; and Sokolova, M. 2015. A framework for a privacy-aware feature selection evaluation measure. In *13th Annual Conference on Privacy, Security and Trust (PST)*, 62–69.

Jolliffe, I. T. 1986. *Principal Component Analysis*. Springer-Verlag.

Miller, A. 2002. *Subset Selection in Regression*. Chapman & Hall/CRC, second edition.

Pattuk, E.; Kantarcioglu, M.; Ulusoy, H.; and Malin, B. 2015. Privacy-aware dynamic feature selection. In *IEEE 31st International Conference on Data Engineering (ICDE)*, 78–88.

Sarwate, A., and Chaudhuri, K. 2013. Signal processing and machine learning with differential privacy: theory, algorithms, and challenges. *IEEE Signal Processing Magazine* 30(5):86–94.

Tramèr, F.; Zhang, F.; Juels, A.; Reiter, M. K.; and Ristenpart, T. 2016. Stealing machine learning models via prediction apis. *CoRR* abs/1609.02943.

Tsoumakas, G.; Spyromitros-Xioufis, E.; Vilcek, J.; and Vlahavas, I. P. 2011. MULAN: A Java Library for Multi-Label Learning. *Journal of Machine Learning Research* 12:2411–2414.

Whitehill, J., and Movellan, J. 2012. Discriminately decreasing discriminability with learned image filters. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2488–2495. IEEE.

Zhang, M., and Zhou, Z. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering* 26(8).