Supplementary Materials for

The reusable holdout: Preserving validity in adaptive data analysis

Cynthia Dwork,* Vitaly Feldman,* Moritz Hardt,* Toniann Pitassi,* Omer Reingold,* Aaron Roth*

*Corresponding author. E-mail: dwork@microsoft.com (C.D.); vitaly@post.harvard.edu (V.F.); m@mrtz.org (M.H.); toni@cs.toronto.edu (T.P.); omer.reingold@gmail.com (O.R.); aaroth@cis.upenn.edu (A.R.)

Published 7 August 2015, *Science* **349**, 636 (2015)
DOI: 10.1126/science.aaa9375

This PDF file includes:

Other Supplementary Material for this manuscript includes the following:
(available at www.sciencemag.org/content/349/6248/636/suppl/DC1)

Supplemental Material

Contents:

1 Differential Privacy, Max Information, and Adaptive Data Analysis

1.1 Our Focus on Linear Statistics

While our approach can be applied to any output of adaptive data analysis (see (*19*) for more details), in our exposition we focus on linear statistics, that is, estimators of the expected value of some function $\phi\colon \mathcal{X} \to [0,1]$ on the distribution $\mathcal{P}$ (also referred to as a linear functional of $\mathcal{P}$ in this context).

We make this choice for three reasons. First, a variety of quantities of interest in data analysis can be expressed as the expectation $\mathcal{P}[\phi] = \mathbb{E}_{x\sim\mathcal{P}}\,\phi(x)$ of some function on $\mathcal{P}$. For example, true means and moments of individual attributes, correlations between attributes and the generalization error of a predictive model; moreover, sufficiently precise estimations of these expectations suffice for model selection and assessment. Next, a request for an approximation to the expectation of a bounded function on $\mathcal{X}$ is referred to as a statistical query in the context of the well-studied statistical query model (*20*), and it is known that using statistical queries in place of direct access to data it is possible to implement most standard analyses used on i.i.d. data (see (*20, 21*) for examples). Thus, a differentially private algorithm for answering many adaptively chosen statistical queries is useful for both settings discussed above: implementing a reusable holdout while avoiding overfitting to the holdout set; and arbitrary adaptive analysis of the entire dataset. Finally, the problem of providing accurate answers to a large number of queries for the average value of a function on the dataset has been the subject of intense investigation in the differential privacy literature. The average value of a function $\phi : \mathcal{X} \to [0,1]$ on a set of random samples is the standard estimator of $\mathcal{P}[\phi]$. In the differential privacy literature such queries are referred to as (fractional) counting queries.

A dataset S consists of n samples drawn randomly and independently from some unknown distribution $\mathcal{P}$ over a discrete universe $\mathcal{X}$ of possible data points. Whether for testing hypotheses or for more general adaptive data analysis, the analyst needs an accurate estimate of $\mathcal{P}[\phi]$ for ϕ of her choice or, equivalently, a confidence interval of small width and high coverage rate (see Sec. 4 for a description of the connection to confidence intervals on linear functionals). Given a dataset $S = (x_1, \ldots, x_n)$, a natural estimator of $\mathcal{P}[\phi]$ is the empirical average $\frac{1}{n}\sum_{i=1}^{n}\phi(x_i)$. We let $\mathcal{E}_S$ denote the empirical distribution that assigns weight $1/n$ to each of the data points in S and thus $\mathcal{E}_S[\phi]$ is equal to the empirical average of ϕ. The standard Hoeffding bound implies that for a fixed function (chosen independently of the data) the probability over the choice of the dataset that this estimator has error greater than τ is at most $2\cdot\exp(-2\tau^2 n)$. This implies that an exponential in n number of functions can be evaluated within τ as long as the functions do not depend on the data.

Our goal is to design a mechanism for accurately estimating the expectations of adaptively chosen functions, i.e., we seek to neutralize the risk of overfitting to S. In the reusable holdout application, access to the training data is unrestricted, and our aim is to prevent overfitting to the holdout set, so in that application the dataset S refers only to the holdout set.

1.2 Generalization via Max Information

Our approach is based on the central insight of the differentially private data analysis: it is possible to learn statistical properties of a dataset while controlling the amount of information "leaked" about any dataset element. We take the same view of the adaptive data reuse problem: the analyst can be prevented from overfitting to the data if the amount of information about the data leaked to the analyst is limited. Information leakage can be limited by controlling the access of the analyst to the data. To quantify how much information has been learned about the data by the analyst we introduce the notion of maximum information. Formally, for jointly distributed random variables $(\boldsymbol{X}, \boldsymbol{Y})$, the next definition bounds the logarithm of the factor by which uncertainty about $\boldsymbol{X}$ is reduced given the value of $\boldsymbol{Y}$.

Definition 1. *Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be jointly distributed random variables. The max-information between $\boldsymbol{X}$ and $\boldsymbol{Y}$, denoted $I_\infty(X;Y)$, is the the minimal value of k such that for every x in the support of $\boldsymbol{X}$ and y in the support of $\boldsymbol{Y}$ we have $\mathbb{P}[\boldsymbol{X} = x \mid \boldsymbol{Y} = y] \leq 2^k \mathbb{P}[\boldsymbol{X} = x]$.*

In our use $(\boldsymbol{X}, \boldsymbol{Y})$ is going to be a joint distribution $(\boldsymbol{S}, \boldsymbol{\phi})$ on (dataset, function) pairs. The dataset $\boldsymbol{S}$ is drawn from distribution $\mathcal{P}^n$ that corresponds to n points drawn i.i.d. from $\mathcal{P}$. Random variable $\boldsymbol{\phi}$ represents the function generated by the analyst while interacting with $\boldsymbol{S}$ through our mechanism. Importantly, the analyst may arrive at the function $\boldsymbol{\phi}$ after observing the evaluations of other functions on the same dataset S. Now with each possible function ϕ in the support of $\boldsymbol{\phi}$ we associate a set of "bad" datasets $R(\phi)$. We later choose $R(\phi)$ to mean the empirical value $\mathcal{E}_S[\phi]$ is far from the true value $\mathcal{P}[\phi]$, that is ϕ overfits to S . Maximum information gives a bound on the probability that $\boldsymbol{S}$ falls in $R(\boldsymbol{\phi})$.

Theorem 2. *For $k = I_\infty(\boldsymbol{S}; \boldsymbol{\phi})$, $\mathbb{P}[\boldsymbol{S} \in R(\boldsymbol{\phi})] \leq 2^k \cdot \max_\phi \mathbb{P}[\boldsymbol{S} \in R(\phi)]$.*

Proof. Definition 1 requires that for all S in support of $\boldsymbol{S}$ and ϕ in support of $\boldsymbol{\phi}$: $\mathbb{P}[\boldsymbol{S} = S \mid \boldsymbol{\phi} = \phi] \leq 2^k \mathbb{P}[\boldsymbol{S} = S]$. Therefore,

$$\begin{aligned}
\mathbb{P}[\boldsymbol{S} \in R(\boldsymbol{\phi})] &= \sum_\phi \mathbb{P}[\boldsymbol{S} \in R(\phi) | \boldsymbol{\phi} = \phi] \, \mathbb{P}[\boldsymbol{\phi} = \phi] \\
&= \sum_\phi \mathbb{P}[\boldsymbol{\phi} = \phi] \sum_{S \in R(\phi)} \mathbb{P}[\boldsymbol{S} = S | \boldsymbol{\phi} = \phi] \\
&\leq \sum_\phi \mathbb{P}[\boldsymbol{\phi} = \phi] \sum_{S \in R(\phi)} 2^k \, \mathbb{P}[\boldsymbol{S} = S] \\
&= 2^k \sum_\phi \mathbb{P}[\boldsymbol{\phi} = \phi] \sum_{S \in R(\phi)} \mathbb{P}[\boldsymbol{S} = S] \\
&= 2^k \sum_\phi \mathbb{P}[\boldsymbol{\phi} = \phi] \, \mathbb{P}[\boldsymbol{S} \in R(\phi)] \leq 2^k \max_\phi \mathbb{P}[\boldsymbol{S} \in R(\phi)].
\end{aligned}$$

□

Our theorem is completely general in the sense that the random variable $\boldsymbol{\phi}$ does not have to be supported on functions over $\mathcal{X}$ and could instead assume values in any other discrete domain.

For example, such output could be a set of features of the data to be used for a subsequent supervised learning task. For our main application ϕ refers to a function, and we denote the set of datasets on which the empirical estimator has error greater than τ as

$$R_\tau(\phi) = \{S \in \mathcal{X}^n \colon \mathcal{E}_S[\phi] - \mathcal{P}[\phi] > \tau\}. \tag{1}$$

By Hoeffding's bound we know that $\max_\phi \mathbb{P}[\boldsymbol{S} \in R_\tau(\phi)] \leq \exp(-2\tau^2 n)$. This gives the following immediate corollary.

Corollary 3. *If $I_\infty(\boldsymbol{S}; \boldsymbol{\phi}) \leq \log_2 e \cdot \tau^2 n$, then $\mathbb{P}[\boldsymbol{S} \in R_\tau(\boldsymbol{\phi})] \leq \exp(-\tau^2 n)$.*

1.3 Differential Privacy Bounds Max Information

We now formally introduce differential privacy and show that it gives one possible way to bound the value of max information. On an intuitive level, differential privacy hides the data of any single individual. We are thus interested in pairs of datasets x, y that differ in a single element, in which case we say x and y are adjacent.

Definition 4. *(13, 22) A randomized algorithm $\mathcal{M}$ with domain $\mathcal{X}^n$ is (ϵ, δ)-differentially private if for all $\mathcal{S} \subseteq \mathrm{Range}(\mathcal{M})$ and for all pairs of adjacent datasets $x, y \in \mathcal{X}^n$:*

$$\mathbb{P}[\mathcal{M}(x) \in \mathcal{S}] \leq \exp(\epsilon)\, \mathbb{P}[\mathcal{M}(y) \in \mathcal{S}] + \delta,$$

where the probability space is over the coin flips of the algorithm $\mathcal{M}$. The case when $\delta = 0$ is sometimes referred to as pure differential privacy, and in this case we may say simply that $\mathcal{M}$ is ϵ-differentially private.

We now show that pure differential privacy implies a bound on max information $I_\infty(\boldsymbol{S}; \boldsymbol{Y})$.

Lemma 5. *Let $\mathcal{M}$ be an ϵ-differentially private algorithm. Let $\boldsymbol{S}$ be any random variable over n-element input datasets for $\mathcal{M}$ and let $\boldsymbol{Y}$ be the corresponding output distribution $\boldsymbol{Y} = \mathcal{M}(\boldsymbol{S})$. Then $I_\infty(\boldsymbol{S}; \boldsymbol{Y}) \leq \log_2 e \cdot \epsilon n$.*

Proof. We will argue that $I_\infty(\boldsymbol{Y}; \boldsymbol{S}) \leq \log_2 e \cdot \epsilon n$; that $I_\infty(\boldsymbol{S}; \boldsymbol{Y}) \leq \log_2 e \cdot \epsilon n$ follows immediately from the Bayes' rule. Clearly, any two datasets x and y differ in at most n elements. Therefore, for every y we have $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = x] \leq e^{\epsilon n}\, \mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = y]$ (this is a direct implication of Definition 4 referred to as group privacy (*23*)). Since there must exist a dataset y such that $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = y] \leq \mathbb{P}[\boldsymbol{Y} = y]$ we can conclude that for every x and every y it holds that $\mathbb{P}[\boldsymbol{Y} = y \mid \boldsymbol{S} = x] \leq e^{\epsilon n}\, \mathbb{P}[\boldsymbol{Y} = y]$. This yields $I_\infty(\boldsymbol{Y}; \boldsymbol{S}) \leq \log_2 e \cdot \epsilon n$ and concludes the proof. □

From Lemma 5 and Corollary 3 we see that ensuring τ^2-differential privacy over the entire interaction with the dataset strictly controls the probability that the adversary can choose a function that overfits to the dataset.

In (*19*) we show that by considering a simple relaxation of max-information, referred to as approximate max-information, it is possible to prove stronger bounds on max-information of differentially private algorithms for datasets consisting of i.i.d. samples. In addition, approximate max-information can be bounded for algorithms whose output has a short description length in bits.

1.4 Stronger Bounds for Differentially Private Algorithms

While Section 1.3 illustrates the key idea of our approach, it requires a relatively strong privacy parameter $\epsilon = \tau^2$. A direct use of differential privacy allows us to reduce the requirement to $\epsilon = \tau$ and also to use (ϵ, δ)-differential privacy with $\delta > 0$. This is summarized in the next theorems, where the probability that a function overfits a random dataset is denoted β; recall that in our application $\beta = e^{-2\tau^2 n}$.

Theorem 6. *Let $\mathcal{M}$ be an ϵ-differentially private algorithm and let $\boldsymbol{S}$ be a random variable drawn from a distribution $\mathcal{P}^n$ ranging over $\mathcal{X}^n$. Let $\boldsymbol{Y} = \mathcal{M}(\boldsymbol{S})$ be the corresponding output distribution. Assume that for each element y in the range of $\mathcal{M}$ there is a subset $R(y) \subseteq \mathcal{X}^n$ so that $\max_y \mathbb{P}[\boldsymbol{S} \in R(y)] \leq \beta$. Then, for $\epsilon \leq \sqrt{\frac{\ln(1/\beta)}{2n}}$ we have $\mathbb{P}[\boldsymbol{S} \in R(\boldsymbol{Y})] \leq 3\sqrt{\beta}$.*

For the special case where the algorithm outputs a function from $\mathcal{X}$ to $[0, 1]$ we may take $\beta = e^{-2\tau^2 n}$ to conclude:

Corollary 7. *Let $\mathcal{M}$ be an ϵ-differentially private algorithm that outputs a function from $\mathcal{X}$ to $[0, 1]$. For a random variable $\boldsymbol{S}$ distributed according to $\mathcal{P}^n$ we let $\boldsymbol{\phi} = \mathcal{M}(\boldsymbol{S})$. Then for any $\tau > 0$, setting $\epsilon \leq \tau$ ensures $\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}]| > \tau\right] \leq 6 \cdot e^{-\tau^2 n}$.*

Relaxing to (ϵ, δ)-differential privacy gives us a wider class of algorithms on which to draw, including some with better bounds on the number of queries that can be answered than is achievable with pure differential privacy:

Theorem 8. *Let $\mathcal{M}$ be an (ϵ, δ)-differentially private algorithm that outputs a function from $\mathcal{X}$ to $[0, 1]$. For a random variable $\boldsymbol{S}$ distributed according to $\mathcal{P}^n$ we let $\boldsymbol{\phi} = \mathcal{M}(\boldsymbol{S})$. Then for any $\tau > 0$ and $n \geq 48 \ln(8/\beta)/\tau^2$, setting $\epsilon \leq \tau/4$ and $\delta \leq (\beta/8)^{4/\tau}$ ensures $\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}] - \mathcal{E}_{\boldsymbol{S}}[\boldsymbol{\phi}]| > \tau\right] \leq \beta$.*

We note that the constant factors in the results above have not been optimized and the bounds are given primarily to demonstrate a qualitatively new dependence on the parameters. We provide the proofs of these statements in (*24*).

2 Details of Thresholdout

Below we present the formal details of Thresholdout and the guarantees that it enjoys.

Algorithm Thresholdout
Input: Training set S_t, holdout set S_h, threshold T, tolerance τ, budget B
Query step: Set $\hat{T} \leftarrow T + \gamma$ for $\gamma \sim \mathrm{Lap}(4 \cdot \tau)$. Given a function $\phi\colon \mathcal{X} \to [-1, 1]$, do:

1. If $B < 1$ output "$\perp$"
2. Else sample $\xi \sim \mathrm{Lap}(2 \cdot \tau)$, $\gamma \sim \mathrm{Lap}(4 \cdot \tau)$, and $\eta \sim \mathrm{Lap}(8 \cdot \tau)$
   (a) If $|\mathcal{E}_{S_h}[\phi] - \mathcal{E}_{S_t}[\phi]| > \hat{T} + \eta$, output $\mathcal{E}_{S_h}[\phi] + \xi$ and set $B \leftarrow B - 1$ and $\hat{T} \leftarrow T + \gamma$.
   (b) Otherwise, output $\mathcal{E}_{S_t}[\phi]$.

Fig. S1. **The details of** Thresholdout **algorithm.**

Note the seeming discrepancy between the guarantee provided by Thresholdout and Theorem 8: while Theorem 8 promises generalization bounds for functions that are generated by a differentially private algorithm, here we allow an arbitrary data analyst to generate query functions in any way she chooses, with access to the training set and differentially private estimates of the means of her functions on the holdout set. The connection comes from the following important property of differential privacy, known as preservation of its guarantee under post-processing (*23, Prop. 2.1*):

Lemma 9. *If $\mathcal{A}$ is an (ϵ, δ)-differentially private algorithm with domain $\mathcal{X}^n$ and range $\mathcal{O}$, and $\mathcal{B}$ is any, possibly randomized, algorithm with domain $\mathcal{O}$ and range $\mathcal{O}'$, then the algorithm $\mathcal{B} \circ \mathcal{A}$ with domain $\mathcal{X}^n$ and range $\mathcal{O}'$ is also (ϵ, δ)-differentially private.*

Towards proving generalization guarantees of Thresholdout we first state what privacy parameters are achieved by Thresholdout.

Lemma 10. Thresholdout *satisfies $(B/\tau n, 0)$-differential privacy.* Thresholdout *also satisfies $(\sqrt{8B \ln(2/\delta)}/\tau n, \delta)$-differential privacy for any $\delta > 0$.*

Consider the first guarantee of Lemma 10. In order to achieve generalization error τ via Corollary 7 (i.e. in order to guarantee that for every function ϕ we have: $\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}] - \mathcal{E}_{\boldsymbol{S}_h}[\boldsymbol{\phi}]| > \tau\right] \leq 6 \cdot e^{-\tau^2 n}$) we need to set the budget so that we achieve $(\epsilon, 0)$-differential privacy for $\epsilon \leq \tau$. To this end we use the following budget function $\mathrm{budget}_0(n, \tau) = \tau^2$. We can also make use of the second guarantee together with Theorem 8. In order to achieve generalization error τ with probability $1 - \beta$ (i.e. in order to guarantee for every function ϕ we have: $\mathbb{P}\left[|\mathcal{P}[\boldsymbol{\phi}] - \mathcal{E}_{\boldsymbol{S}_h}[\boldsymbol{\phi}]| > \tau\right] \leq \beta$), we can apply Theorem 8 by setting $\epsilon = \tau/4$ and $\delta = (\beta/8)^{4/\tau}$. We can obtain these privacy parameters from Lemma 10 by setting the budget as follows:

$$\mathrm{budget}_1(n, \tau, \beta) = \frac{\tau^5 n^2}{512 \ln(8/\beta)}\,. \tag{2}$$

Both settings lead to small generalization error and so we can pick whichever gives the larger bound. The first bound has a milder dependence on τ, but grows only linearly with n.

The second bound features a quadratic dependence on n at the expense of a worse dependence on the other parameters. In the regime in which the size of the dataset n grows large, the second bound will eventually dominate the first bound.

We can now apply our main results to get a generalization bound for Thresholdout:

Theorem 11. *Let $T \geq 0, \tau > 0$. Let S_h and S_t be data sets drawn i.i.d. from a distribution $\mathcal{P}$. Consider an algorithm that is given access to S_t and adaptively chooses functions $\phi_1, \ldots, \phi_m$ while interacting with* Thresholdout *which is given data sets S_h, S_t tolerance τ, and threshold T, and returns an answer a_i on function $\phi_i : \mathcal{X} \to [0, 1]$. If we set $B = \mathrm{budget}_0(|S_h|, \tau)$, then, for all $i \in \{1, \ldots, m\}$ such that $a_i \neq \perp$, we have for all $t > 0$*

$$\mathbb{P}\left\{|a_i - \mathcal{P}[\phi_i]| > T + (t+1)\tau\right\} \leq 6 \cdot e^{-\tau^2 n} + e^{-t/8}\,.$$

If we put $B = \mathrm{budget}_1(|S_h|, \tau, \beta)$ then for all $i \in \{1, \ldots, m\}$ such that $a_i \neq \perp$ we have for all $t > 0$

$$\mathbb{P}\left\{|a_i - \mathcal{P}[\phi_i]| > T + (t+1)\tau\right\} \leq \beta + e^{-t/8}\,.$$

The conclusion of the theorem shows that the generalization error of the algorithm is tightly concentrated around $T + \tau$. We typically choose the parameters T and τ so that $T + \tau$ is reasonably small, e.g., 0.05. Increasing T relative to τ reduces the number of functions with above threshold response, but increases the generalization error. Decreasing T has the opposite effect and even $T = 0$ leads to a non-trivial guarantee. Proofs of Lemma 10 and Theorem 11 are given in (*19*).

3 Additional Details on the Experiment

Implementation details of Thresholdout

We used an implementation of Thresholdout that differs somewhat from the algorithm we analyzed theoretically (given in Fig. S1). Specifically, we use an idealized setting of the budget function that ignores some of the overhead present in our theoretical analysis. Second, we use Gaussian noise instead of Laplacian noise as it has stronger concentration guarantees leading to a more economic use of the budget.

Implementation of the Linear Classifier

The algorithm implemented by the analyst in our experiment is given n labeled examples over $\mathbb{R}^d$. In the first step it selects k variables and in the second one it builds a linear threshold classifier.

1. For each attribute $i \in [d]$ compute the correlation with the label on the training and holdout sets: $w_i^t = \sum_{(x,y)\in S_t} x_i y$ and $w_i^h = \sum_{(x,y)\in S_h} x_i y$. Let

$$W = \left\{i \mid w_i^t \cdot w_i^h > 0;\ |w_i^t| \geq 1/\sqrt{n};\ |w_i^h| \geq 1/\sqrt{n}|\right\}$$

that is the set of variables for which w_i^t and w_i^h have the same sign and both are at least $1/\sqrt{n}$ in absolute value (this is the standard deviation of the correlation in our setting). Let V_k be the subset of variables in W with k largest values of $|w_i^t|$.

2. Construct the classifier $f(x) = \mathsf{sign}\left(\sum_{i \in V_k} \mathsf{sign}(w_i^t) \cdot x_i\right)$.

Accuracy on the Holdout Set

In simulations that used Thresholdout for selecting the variables we have also plotted the accuracy on the holdout set as reported by Thresholdout. For comparison purposes, in Fig. S2 we plot the actual accuracy of the generated classifier on the holdout set (the parameters of the simulation are identical to those used in Figs. 1B and 2B). It demonstrates that there is essentially no overfitting to the holdout set. Note that the advantage of the accuracy reported by Thresholdout is that it can be used to make further data dependent decisions (such as selecting the best classifier to output) without risk of overfitting.

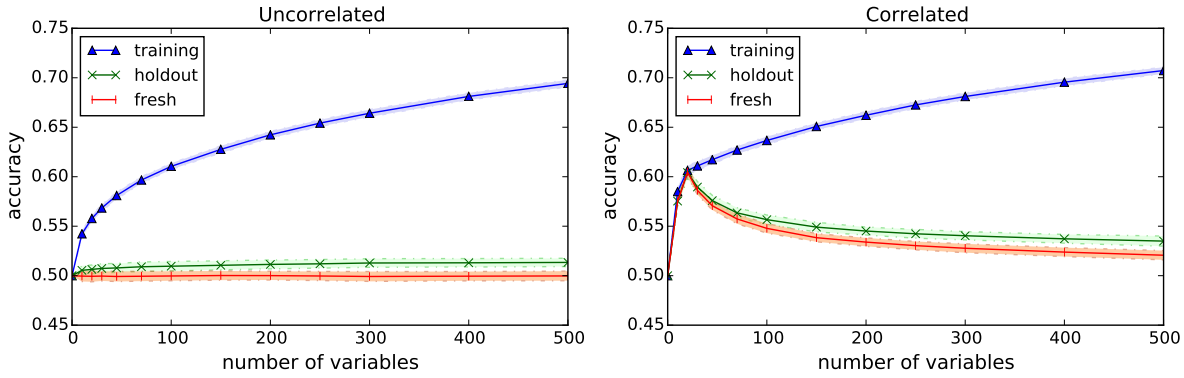

Fig. S2. Accuracy of the classifier produced with Thresholdout on the holdout set.

4 Connection to Confidence Intervals

Below we briefly explain how the guarantees of our algorithm can be viewed as a conservative confidence interval on a linear (statistical) functional. Recall that $T(\mathcal{P})$ is a linear functional if $T(\mathcal{P}) = \mathbb{E}_{x \sim \mathcal{P}}[\phi(x)]$ for some real-valued function $\phi : \mathcal{X} \to \mathbb{R}$. See (*25, Sec. 7.2*) for a more detailed definition and a discussion of standard confidence intervals on a linear functional.

For concreteness, consider a simple example in which $\mathcal{X} = \{0, 1\}$ and the distribution $\mathcal{P}$ is a Bernoulli distribution with (unknown) success probability p, denoted by $B(p)$. Let $x_1, \ldots, x_n$ denote the n given samples and let $I(x_1, \ldots, x_n) \subseteq [0, 1]$ be some interval constructed given

the data points. I is said to be a conservative confidence interval for p with coverage probability $1-\beta$ if for every $p \in [0,1]$,

$$\mathbb{P}_{x_1,\ldots,x_n \sim B(p)}[p \in I(x_1,\ldots,x_n)] \geq 1-\beta.$$

The standard way to construct such an interval is to take an interval around the sample mean $\hat{p} = \frac{1}{n}\sum_{i=1}^{n} x_i$. Various ways exist to pick the width of the interval on the basis of n, β and $\hat{p}$. A common (if imprecise) choice is to use the normal approximation to the binomial distribution. Namely, set $I = [\hat{p}-\alpha, \hat{p}+\alpha]$, where $\alpha = z_{\beta/2} \cdot \sqrt{\hat{p}(1-\hat{p})/n}$ and $z_{\beta/2}$ is the value satisfying $\mathbb{P}_{Z\sim N(0,1)}[Z \geq z_{\beta/2}] = \beta/2$. By the properties of the normal distribution, we know that α scales linearly with $\sqrt{\ln(1/\beta)/n}$.

Algorithms that we describe have the following guarantee: given a dataset $S = (x_1,\ldots,x_n)$ and two values α and β chosen by the analyst, an algorithm A, given a function $\phi : \mathcal{X} \to [0,1]$, outputs a value v such that

$$\mathbb{P}_{A;\ x_1,\ldots,x_n \sim P}[|\ \mathop{\mathbb{E}}_{x\sim\mathcal{P}}[\phi(x)] - v| > \alpha] \leq \beta. \tag{3}$$

Going back to the Bernoulli example, let us apply this guarantee to a function $\phi(x) = x$. Then $\mathbb{E}_{x\sim B(p)}[\phi(x)] = \mathbb{E}_{x\sim B(p)}[x] = p$. Note that the statement (3) is then exactly equivalent to the following: for every $p \in [0,1]$,

$$\mathbb{P}_{A;\ x_1,\ldots,x_n \sim B(p)}[p \in [v-\alpha, v+\alpha]] \geq 1-\beta.$$

In other words, the guarantee of the algorithm A is that for the estimate v it outputs, the interval $[v-\alpha, v+\alpha]$ is a conservative confidence interval with coverage rate $1-\beta$. One small difference is that our algorithm (or estimator) is randomized (that is, depends both on data and internal randomness of A) and therefore the probability statement is also over the randomness of A. Many statistical procedures are randomized so this is a natural formulation of confidence intervals for such procedures (for example randomly splitting samples in two parts introduces randomization into the output of a procedure). Chernoff-Hoeffding concentration inequalities imply that for estimating the success probability of a single such variable we need n that scales linearly with $\ln(1/\beta)/\alpha^2$ to solve this task or, equivalently, α scales linearly with $\sqrt{\ln(1/\beta)/n}$ (as in the normal approximation above).

Extending on this simple example, consider a $\{0,1\}$-valued function ϕ over some universe $\mathcal{X}$ and the associated linear functional $\mathbb{E}_{\mathcal{P}}[\phi]$. For a Boolean function ϕ, the value of ϕ on x chosen randomly from some distribution $\mathcal{P}$ over $\mathcal{X}$ is a Bernoulli random variable with success probability $\mathbb{E}_{\mathcal{P}}[\phi]$. Therefore constructing a confidence interval for the linear functional $\mathbb{E}_{\mathcal{P}}[\phi]$ is, in essence, just the same task as constructing a confidence interval on the Bernoulli success probability of a $\{0,1\}$-valued random variable defined by the analyst. In particular, as in the example above, the guarantees of our algorithm (given in eq.(3)) give a conservative confidence interval on the linear functional $\mathbb{E}_{\mathcal{P}}[\phi]$.

The generality of our results comes from the ability to pick any sequence of arbitrary functions ϕ on the universe $\mathcal{X}$. This, as is well-recognized in machine learning theory, allows many different analyses to be performed on the data (e.g. (*21*)).

For a qualitative understanding of our bounds consider the problem of constructing confidence intervals for m adaptively chosen linear functionals. Essentially, the only previously known solution to this problem with rigorous guarantees would be to use fresh samples for every linear functional (again, in the adaptive setting functionals depend on data so a naïve application of standard confidence intervals is likely to lead to invalid results). Therefore, given n samples, one would use n/m samples to give a confidence interval for each linear functional resulting in confidence interval width α which is roughly proportional to $\sqrt{m \ln(1/\beta)/n}$. For comparison, our bounds in Thm. 11, for an appropriate choice of values T and τ, give α which is proportional to $\ln(m/\beta)\sqrt{B/n}$, where B is the number of times our procedure detects and corrects overfitting to the training dataset. Note that the crucial dependence on m is now exponentially better than in the known approach (as long as B is small relative to n).

Data S1 (.py file). Python code for the numerical experiments reported in Figures 1, 2, and S2.