Knowl Inf Syst (2018) 54:95–122
https://doi.org/10.1007/s10115-017-1116-3

REGULAR PAPER

Auditing black-box models for indirect influence

Philip Adler[1] · Casey Falk[1] · Sorelle A. Friedler[1] · Tionney Nix[1] · Gabriel Rybeck[1] · Carlos Scheidegger[2] · Brandon Smith[1] · Suresh Venkatasubramanian[3]

Received: 1 April 2017 / Revised: 22 August 2017 / Accepted: 10 October 2017 /
Published online: 25 October 2017

Abstract Data-trained predictive models see widespread use, but for the most part they are used as *black boxes* which output a prediction or score. It is therefore hard to acquire a deeper understanding of model behavior and in particular how different features influence the model prediction. This is important when interpreting the behavior of complex models or asserting that certain problematic attributes (such as race or gender) are *not* unduly influencing decisions. In this paper, we present a technique for *auditing* black-box models, which lets us study the extent to which existing models take advantage of particular features in the data set, without knowing how the models work. Our work focuses on the problem of *indirect influence*: how some features might indirectly influence outcomes via other, related features. As a result, we can find attribute influences even in cases where, upon further direct examination of the model, *the attribute is not referred to by the model at all.* Our approach does not require the black-box model to be retrained. This is important if, for example, the model is only accessible via an API, and contrasts our work with other methods that investigate feature influence such as feature selection. We present experimental evidence for the effectiveness of our procedure using a variety of publicly available data sets and models. We also validate our procedure using techniques from interpretable learning and feature selection, as well as against other black-box auditing procedures. To further demonstrate the effectiveness of this technique, we use it to audit a black-box recidivism prediction algorithm.

A preliminary version of this work with authors Philip Adler, Casey Falk, Sorelle A. Friedler, Gabriel Rybeck, Carlos Scheidegger, Brandon Smith, and Suresh Venkatasubramanian was titled *Auditing Black-box Models for Indirect Influence* and appeared in the *Proceedings of the IEEE International Conference on Data Mining (ICDM)* in 2016. This research was funded in part by the NSF under Grants IIS-1251049, CNS-1302688, IIS-1513651, DMR-1307801, IIS-1633724, and IIS-1633387.

✉ Sorelle A. Friedler
sorelle@cs.haverford.edu

[1] Department of Computer Science, Haverford College, Haverford, PA 19041, USA

[2] Department of Computer Science, University of Arizona, Tucson, AZ, USA

[3] Department of Computer Science, University of Utah, Salt Lake City, UT, USA

Keywords Black-box auditing · ANOVA · Algorithmic accountability · Deep learning · Discrimination-aware data mining · Feature influence · Interpretable machine learning

1 Introduction

Machine learning models now determine and control an increasing number of real-world decisions, from sentencing guidelines and parole hearings [3] to predicting the outcome of chemical experiments [24]. These models, powerful as they are, tend to also be opaque. This presents a challenge. How can we *audit* such models to understand why they make certain decisions? As conscientious model creators, we should want to know the extent to which a specific feature contributes to the accuracy of a model. As outside auditors, trying to understand a system can give us an understanding of the model's priorities and how it is influenced by certain features. This may even have legal ramifications: by law, for example, decisions about hiring cannot be influenced by factors such as race, gender or age.

As model creators, we could build interpretable models, either by explicitly using interpretable structures such as decision trees or by building shadow models that match model outputs in an interpretable way. In this work, we are interested in auditing a *black-box* model from the outside (because the model is proprietary, accessible only through an API or cannot be modified).

1.1 Direct and indirect influence

Much of the modern literature on black-box auditing (see Sect. 2 for details) focuses on what we call *direct* influence: how does a feature (or a group of features) directly affect the outcome? This is quantified by replacing the feature (or group) by random noise and testing how model accuracy deteriorates. In this paper, we focus on the different and more subtle issue of *indirect* influence.

Consider trying to verify that racial considerations did *not* affect an automated decision to grant a housing loan. We could use a standard auditing procedure that declares that the `race` attribute does not have an undue influence over the results returned by the algorithm. Yet this may be insufficient. In the classic case of *redlining* [21], the decision-making process explicitly excluded race, but it used `zipcode`, which in a segregated environment is strongly linked to race. Here, race had an *indirect* influence on the outcome via the zipcode, which acts as a *proxy*.

Note that in this setting, race would not be seen as having a direct influence (because removing it doesn't remove the signal that it provides). Removing both race and zipcode jointly (as some methods propose to do) reveals their combined influence, but also eliminates other task-specific value that the zipcode might signal independent of race, as well as leaving unanswered the problem of *other* features that race might exert indirect (and partial) influence through.

1.2 Our work

In this paper, we study the problem of auditing black-box models for indirect influence. In order to do this, we must find a way to capture information flow from one feature to another. We take a learning theoretic perspective on this problem, which can be summarized via the principle, first enunciated in [13] in the context of certifying and removing bias in classifiers:

the information content of a feature can be estimated by trying to predict *it from the remaining features*.

How does this allow us to correctly quantify the influence of a feature in the presence of proxies? Let us minimally modify the data so that the feature can no longer be predicted from the remaining data. The above principle then argues that we have fully eliminated the influence of this feature, both directly and in any proxy variables that might exist. If we now test our model with this *obscured* data set, any resulting drop in prediction accuracy can be attributed directly to information from the eliminated feature.

Our main contributions in this work include:

- A technique to *obscure* (fully and partially) the influence of a feature on an outcome, and a theoretical justification of this approach.
- A method for quantifying indirect influence based on a differential analysis of feature influence before and after obscuring.
- An experimental validation of our approach on a number of public data sets, using a variety of models.
- A technique for interpreting indirect influence and experimental validation of that approach on a black-box recidivism prediction algorithm.

2 Conceptual context

Work on black-box auditing intersects with a number of related areas in machine learning (and computer science at large), including directions coming from privacy preservation, security, interpretability and feature selection. We tease out these connections in more detail in Sect. 8.

Here we outline how we see our work in the specific context of the prior literature on auditing black-box machine learning models. Modern developments in this area can be traced back to Breiman's work on random forests [5], and we highlight two specific recent related works. Henelius et al. [16] propose looking at variable sets and their influence by studying the consistency of a black-box predictor when compared to a random permutation of a set. Datta et al. [11] provide a generalization of this idea, linking it to game theoretic notions of influence and showing that different choices of probability spaces and random variables yield a number of different interesting auditing measures. These two papers fundamentally hinge on the notion of associating each input value with an *intervention distribution*. These intervention distributions can be easily shown (in distribution) to obscure attributes. Our work, on the other hand, will audit black boxes by providing, for any given input point, an intervention that is *deterministic*, while guaranteeing (in some settings, see Sect. 4.3) that the attributes are still obscured over the entire distribution. Our intervention preserves more of the signal in the data set and—crucially in some settings—naturally preserves indirect influences of proxy variables. As we will show in the experiments (see Sect. 5.5), the technique of Henelius et al. cannot detect proxy variables, and although Datta et al. can use some of their measures to detect proxy variables, their attribute rankings generally do not reflect the proxy relationships.

Our methods draw heavily on ideas from the area of *algorithmic fairness*. The process by which we eliminate the influence of a feature uses ideas from earlier work on testing for disparate impact [13]. Again, a key difference is that we no longer have the ability to retrain the model, and we *quantify* the influence of a feature rather than merely eliminating its influence.

3 Indirect influence

Let $f : \mathbb{X} \to \mathbb{Y}$ be a black-box classification function, where $\mathbb{X} \subset \mathbb{X}^{(1)} \times \mathbb{X}^{(2)} \ldots \mathbb{X}^{(d)}$ is a d-dimensional feature space with the ith coordinate drawn from the domain $\mathbb{X}^{(i)}$ and $\mathbb{Y}$ is the domain of outcomes. For example, $\mathbb{X} = \mathbb{R}^d$ and $\mathbb{Y} = \{-1, 1\}$ for binary classification on Euclidean vectors. Fix a data set $(X, Y) = \{(X_i, y_i)\} \subset \mathbb{X} \times \mathbb{Y}, 1 \leq i \leq n$ and let $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, where $x_{ij} \in \mathbb{X}_j$ for all i. We denote the accuracy of prediction as $\texttt{acc}(X, Y, f)$. For example, $\texttt{acc}(X, Y, f) = \frac{1}{n} \sum \mathbf{1}_{y_i \neq f(X_i)}$ is the standard misclassification error. As usual, let the ℓ_p norm on $\mathbb{R}^d$ be defined as $\|x\|_p = (\sum_{i=1}^{d} |x_i|^p)^{1/p}$.

We wish to quantify the indirect influence of a feature j on the outcome of classification. The typical approach to doing this in the auditing literature is to perturb the jth feature x_{ij} of each X_i in some way (usually by random perturbation), obtaining a modified data set X_{-j}. Then the influence of j can be quantified by measuring the difference between $\texttt{acc}(X, Y, f)$ and $\texttt{acc}(X_{-j}, Y, f)$. (Note that f is not retrained on X_{-j}.)

Unfortunately, randomly perturbing features can disrupt indirect influence in a number of ways. Firstly, random perturbations could also remove useful task-related information in proxy features that would degrade the quality of classification. Secondly, this prevents us from cleanly quantifying the *relative* effect of the feature being perturbed on related proxy variables.

We propose a different approach. We will still perturb a data set X to eliminate the direct and indirect influence of feature j, and measure the change in accuracy of prediction as before. However, we will do this perturbation in a directed and deterministic manner, organized around the question: "*can we predict the value of feature j from the remaining features*?" Intuitively, if we cannot, then we know that we have correctly eliminated the influence of j. Moreover, if we do this perturbation "minimally," then we have changed the data as little as possible in the process. We will say in this case that we have *removed* feature j from the data set and have *obscured* its influence on X.

3.1 Obscuring data with respect to a feature

We start by defining the error measure we will use to test predictability. Rather than the standard misclassification rate, we will use the well-known *balanced error rate* measure that is more sensitive to class imbalance. This is important if a feature has significant skew in the values it takes. Let $\text{supp}(Y) = \{y \in \mathbb{Y} | y \in Y\}$ be the set of elements of $\mathbb{Y}$ that appear in the data.

Definition 3.1 (BER) Let $f : \mathbb{X} \to \mathbb{Y}$ be a classifier, and let $(X, Y) = \{(X_i, y_i)\}$ be a set of examples. The *balanced error rate* BER of f on (X, Y) is the (unweighted) average class-conditioned error of f:

$$\text{BER}(X, Y, f) = \frac{1}{|\text{supp}(Y)|} \left(\sum_{j \in \text{supp}(Y)} \frac{\sum_{y_i = j} \mathbf{1}_{f(X_i) \neq j}}{|\{i \mid y_i = j\}|} \right)$$

A feature i has been removed from a data set if we can no longer predict that feature from the remaining data. This motivates the following definition. Let $X^{(i)} = (x_{1i}, x_{2i}, \ldots, x_{ni})$ denote the column corresponding to the ith feature.

Definition 3.2 (ϵ-obscure) We define $X \setminus_\epsilon \mathbb{X}_i$ as the *ϵ-obscure* version of X with respect to feature $\mathbb{X}_i$ if $X^{(i)}$ cannot be predicted from $X \setminus_\epsilon X_i$. That is, if, for all functions $f : \mathbb{X} \setminus \mathbb{X}_i \to \mathbb{X}_i$,

$$\text{BER}(X \setminus_\epsilon \mathbb{X}_i, X^{(i)}, f) > \epsilon$$

We can now define a measure of influence for a feature.

Definition 3.3 ((indirect) influence) The *indirect influence* II(i) of a feature i on a classifier f applied to data (X, Y) is the difference in accuracy when f is run on X versus when it is run on $X \setminus_\epsilon \mathbb{X}_i$:

$$\text{II}(i) = \texttt{acc}(X, Y, f) - \texttt{acc}(X \setminus_\epsilon \mathbb{X}_i, Y, f)$$

Notes The definition of obscurity we use here is adapted from [13], but applied to any feature, rather than just "protected" ones. In what follows, we will typically treat ϵ as large (say above 0.5 for binary classification).

4 Computing influence

In this section, we will introduce a method we call *gradient feature auditing* (GFA) to estimate indirect influence. Using it, we compute the influence of each feature of the data and order the features based on their influence. This GFA algorithm works feature by feature: in order to compute $X \setminus_\epsilon \mathbb{X}_i$, we apply an *obscuring procedure* to each feature $j \neq i$ in $\mathbb{X}$. Because we operate one feature at a time, we cannot guarantee that all influence can be removed (and therefore estimated). However, we will show that the feature-level procedure is theoretically sound and in fact generalizes the standard ANOVA test for null hypothesis testing.

Let us start with a simple case: when the feature $W = \mathbb{X}_j$ to be obscured is numerical, and the feature $O = \mathbb{X}_i$, we are removing is categorical. Let $W_x = \Pr(W \mid O = x)$ denote the marginal distribution on W conditioned on $O = x$ and let the cumulative distribution be $F_x(w) = \Pr(W \geq w \mid O = x)$.

Define the *median* distribution A such that its cumulative distribution F_A is given by $F_A^{-1}(u) = \text{median}_{x \in O} F_x^{-1}(u)$. In [13], it was shown that if we modify the distribution of W to match A by "moving" values of W so as to mimic the distribution given by A, then O is maximally obscured, but W also minimally changes, in that A also minimizes the function $\sum_{x \in O} d(W_x, A)$ where $d(\cdot, \cdot)$ was the earthmover distance [27] between the distributions using the ℓ_2 distance $d(p, q) = \|p - q\|_2$ as the base metric. We call this procedure `ObscureNumerical`.

This procedure does not work if features to be obscured and removed are not numerical and categorical, respectively. We now describe procedures to address this issue.

4.1 Removing numerical features

In order to remove a numerical feature, we must first determine what aspects of the number itself should be removed. In an optimal setting, we might remove the entirety of the number by considering its binary expansion and ensuring that no bit was recoverable. However, when working with most numerical data, we can safely assume that only the higher-order bits of a number should be removed. For example, when considering measurements of scientific phenomena, the lower-order bits are often measurement error.

Thus, we bin the numerical feature and use the bins as categorical labels in the previously described obscuring procedure. Bins are chosen using the Freedman–Diaconis rule for choosing histogram bin sizes [14].

4.2 Obscuring categorical features

Our procedure relies on being able to compute cumulative density functions for the feature W being obscured. If it is categorical, we no longer have an ordered domain on which to define the cumulative distributions F_w. However, we do have a base metric: the exact metric $\mathbf{1}$ where $\mathbf{1}(x, w) = 1 \iff x = w$. We can therefore define A as before, as the distribution minimizing the function $\sum_{x \in O} d(W_x, A)$. We observe that the earthmover distance between any two distributions over the exact metric has a particularly simple form. Let $p(w), q(w), w \in W, \sum p(w) = \sum q(w) = 1$ be two distributions over W. Then the earthmover distance between p and q with respect to the exact metric $\mathbf{1}$ is given by $d(p, q) = \|p - q\|_1$. Therefore, the desired minimizer A can be found by taking a component-wise median for each value w. In other words, A is the distribution such that $p_A(w) = \text{median}_w W_x(w)$. Once such an A has been computed, we can find the exact repair by computing the earthmover distance (via min-cost flows)[1] between each W_x and A. This results in fewer changes than merely changing values arbitrarily.

We must create the obscured version of W, denoted $\hat{W}$. Let $\hat{W}_{w,x}$ be the partition of $\hat{W}$ where the value of the obscured feature is w and the value of the removed feature is x. We must create $\hat{W}$ so as to ensure that $|\{\hat{W}|O = x\}| \in \mathbb{Z}$ for all values of $x \in O$. We set $|\hat{W}_{w,x}| = \lfloor p_A(w) \cdot |\{W|O = x\}| \rfloor$. Letting $d(w) = |\hat{W}_{w,x}|$ for all $w \in W$ gives the node demands for the circulation problem between W_x and A, where supplies are set to the original counts $d(w) = -|W_{w,x}|$. Since $|\hat{W}_{w,x}| \leq |W_{w,x}|$, an additional *lost observations node* with demand $|W_{w,x}| - |\hat{W}_{w,x}|$ is also added. The flow solution describes how to distribute observations at a per category, per obscured feature value level. Individual observations within these per category, per feature buckets can be distributed arbitrarily. The observations that flow to the lost observations node are distributed randomly according to distribution A. We call this procedure `ObscureCategorical`.

Using the categorical or numerical obscuring procedure appropriately depending on the data yields our procedure for computing $X \setminus_\epsilon \mathbb{X}_i$.

Notes This description assumes that we want to remove *all* effects of a variable in order to measure its influence. However, how the influence changes as we remove its effect is also interesting. Indeed, this is why we refer to the overall process as a *gradient* feature audit. To that end, all of the algorithms above can be adapted to allow for a *partial* removal of a variable. On a scale of 0–1 where 0 represents the original data, and 1 represents a full removal, we can remove a *fractional* part of the influence by allowing the individual conditional distributions to move *partly* toward each other.

While the process above produces a score, the induced ranking is also useful, especially when we compare our results to those produced by other auditing methods, where the score itself might not be directly meaningful. We illustrate this further in Sect. 5.

4.3 Obscuring, ANOVA and the F-test

We now provide a theoretical justification for our obscuring procedure. Specifically, we show that if the feature W being obscured has Gaussian conditionals, then our obscuring procedure will create a data set on which the F-test [7] will fail, which means that the null hypothesis (that the conditionals are now identical) will not be invalidated. Thus, our procedure can be viewed as a generalization of ANOVA.

[1] This is a straightforward application of the standard min-cost flow problem.

Consider a data set D consisting of samples (x, y), where y is a class label that takes the values $-1, 1$ and the x are drawn from univariate Gaussians with different means and a shared variance. Specifically, $\Pr(x|y = i) = \mathcal{N}(v_i, \sigma^2)$. We assume the classes are balanced: $\Pr(y = -1) = \Pr(y = 1)$.

Let $\tilde{x}$ be the obscured version of x. It is easy to show that,[2] in this case, the obscuring procedure will produce values following the distribution $p(\tilde{x}|y = i) = \mathcal{N}(1/2(v_{-1} + v_1), \sigma^2)$.

We apply the F-test to see whether we can tell apart the two conditional distributions for the two different values of y. Let $S_y = \{x \mid (x, y) \in D\}$. The test statistic F is the ratio of the between-group sample variance and the in-group sample variance. Set $\mu = (v_{-1} + v_1)/2$ and $\delta = (\mu - v_{-1})^2 = (\mu - v_1)^2$. Thus,

$$F = \frac{n\delta}{(1/2)\sum_{x \in S_{-1}}(x - v_{-1})^2 + \sum_{x \in S_1}(x - v_1)^2}$$

We note that $\sum_{x \in S_1}(x - v_1)^2$ has expectation $n/2\sigma^2$ (and so does the corresponding expression related to S_{-1}). Using the plug-in principle, we arrive at an estimator for the F statistic: $F = \frac{\delta}{\sigma^2}$. This is the traditional expression for a two-variable, one-way ANOVA: δ is a measure of the variance that is explained by the group, and σ^2 is the unexplained variance.

We apply this to the obscured distribution. From the remarks above, we know that the conditional distributions for $y = 0, 1$ are identical Gaussians $N(\mu, \sigma)$. We now show that the positive parameter δ is concentrated near zero as the number of samples increases.

Let $x_1, \ldots, x_n$ be samples drawn from the conditional distribution for $y = 0$, and similarly let $y_1, \ldots, y_n$ be drawn from the distribution conditioned on $y = 1$. Set $X = \frac{1}{n}\sum x_i$ and $Y = \frac{1}{n}\sum y_i$. Note that $E[X] = E[Y] = \mu$, and so $E[X - Y] = 0$.

Let $\hat{\delta} = (X - Y)^2$. We first observe that

$$\begin{aligned} |X - Y| &\leq |X - E[X]| + |E[X] - E[Y]| + |Y - E[Y]| \\ &= 2|X - E[X]| \end{aligned}$$

because X and Y are identically distributed. Therefore,

$$\begin{aligned} \Pr(\hat{\delta} \geq \epsilon^2) &\leq 4\Pr(|X - E[X]|^2 \geq \epsilon^2) \\ &\leq 4\exp(-n\epsilon^2/2\sigma^2) \end{aligned}$$

by standard Hoeffding tail bounds [22]. Therefore, with $\log n/\epsilon^2$ samples, the F-test statistic is with high probability less than ϵ^2, and thus, the null hypothesis (that the distributions are identical) will not be invalidated (which is equivalent to saying that the test cannot distinguish the two conditional distributions).

5 Experiments

In order to evaluate the introduced gradient feature auditing (GFA) algorithm, we consider experiments on four data sets, chosen to balance easy replicability with demonstration on

[2] This follows from the fact that the earthmover distance between two distributions on the line is the ℓ_1 difference between their cumulative density functions. In this case, it means that the earthmover distance is precisely the distance between the means.

domains where these techniques are of practical interest. Data sets and GFA code are available online.[3]

Synthetic data We generated 6000 items with 3000 assigned to each of two classes. Items have five features. Three features directly encode the row number i: Feature A is i, B is $2i$ and C is $-i$. There is also a random feature and a constant feature. We use a random $\frac{2}{3} : \frac{1}{3}$ training test split.

Adult Income and German Credit data We consider two commonly used data sets from the UC Irvine machine learning repository.[4] The first is the Adult Income data set consisting of 48,842 people, each with 14 descriptive attributes from the US census and a classification of that person as making more or less than $50,000 per year. Missing data are handled using Weka's built-in data imputation. We use the training/test split given in the original data. The second is the German Credit data set consisting of 1000 people, each with 20 descriptive attributes and a classification as having good or bad credit. We use a random $\frac{2}{3} : \frac{1}{3}$ training test split on these data and the data set described below.

Dark Reaction data The Dark Reaction data [24] are a set of 3, 955 historical hydrothermal synthesis experiments aiming to produce inorganic–organic hybrid materials. In total, 273 attributes indicate aggregates of atomic and ionic properties of the inorganic components and semi-empirical properties of the organic components. The classification variable indicates whether the reaction produced an ionic crystal.

Models We built two models that are notoriously opaque to simple examination; SVMs[5] [15] and feedforward neural networks (FNNs).[6] We also include C4.5 decision trees[7] [23] so that the audited results can be examined directly in comparison with the models themselves.

The FNN on the synthetic data was trained using a `softmax` input layer for 100 epochs with a batch size of 500 and a learning rate of 0.01; no hidden layer was involved. The Adult Income FNN model was trained using a single `softmax` input layer for 1000 epochs using a batch size of 300 and a learning rate of 0.01; no hidden layer was involved. The German Credit data FNN model was trained similarly to the Adult model, except using a batch size of 700. The Dark Reaction data FNN model was trained using `tanh` activations for the input layer and `softmax` activations in a fully connected hidden layer of 50 nodes; it was trained using a batch size of 300 for 1500 epochs with a modified learning rate of 0.001.

5.1 Auditing using test data

The work of [13] assumes that $X \setminus X_i$ is used to *train* a function $f : X \setminus X_i \to \hat{Y}$ so that $\hat{Y}$ can be guaranteed to be ϵ-obscured with respect to X_i. However, in a black-box auditing context, we cannot retrain the model with the obscured data set. So is $\hat{Y}$ obscured with respect to X_i if $f : X \setminus X_i \to \hat{Y}$ where $X \setminus X_i$ is the *test* data and the model f was trained on $\mathbb{X}$? We answer this question affirmatively, with empirical justification.

[3] https://github.com/algofairness/BlackBoxAuditing.

[4] https://archive.ics.uci.edu/ml/datasets.html.

[5] Implemented using Weka's version 3.6.13 SMO: http://weka.sourceforge.net/doc.dev/weka/classifiers/functions/SMO.html.

[6] Implemented using TensorFlow version 0.6.0: https://www.tensorflow.org/.

[7] Implemented using Weka's version 3.6.13 J48: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/J48.html.

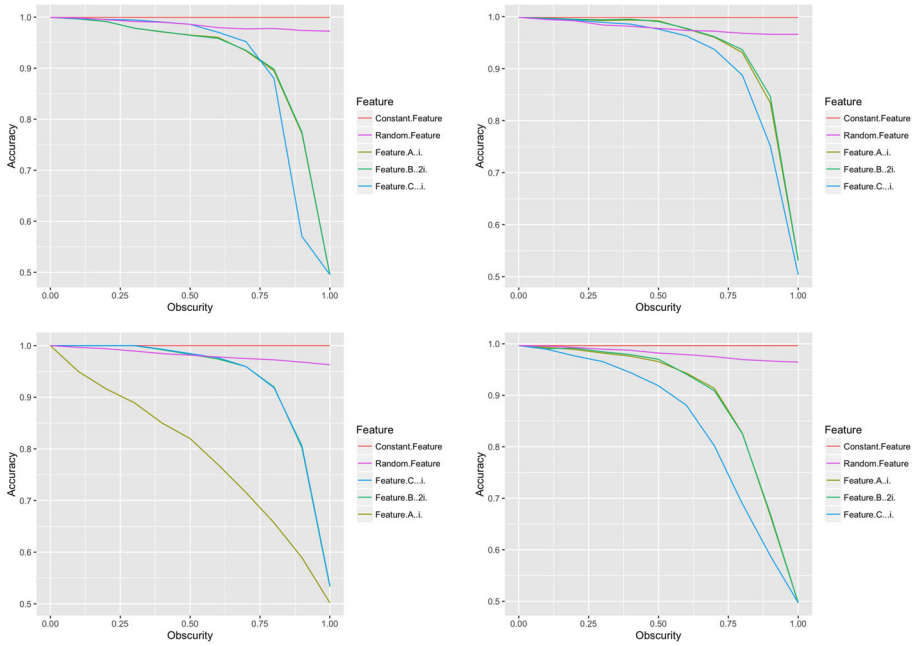


Fig. 1 Synthetic data that are retrained each time the obscuring procedure are run (first row) as compared to the same data with the gradient feature auditing method that obscures only the test data and does not retrain the model. The first column shows this with a decision tree and the second column with an SVM

The intuition behind the obscuring process is that once an attribute W is obscured with respect to an attribute O, there is no way to learn about O given values of W. In [13], this property was used to argue that a model that is *trained* on the obscured data cannot predict O with low error and therefore cannot discriminate.

In a black-box setting, we cannot control how the model was trained. However, we can obscure the *test data*. In that case, even a model trained with knowledge of O cannot take advantage of this information, because the presence of O is masked in the data through the obscuring process. Thus, the results of classification will be free of the influence of O.

We can empirically validate this claim by running an experiment where we *do* retrain the model (a la [13]) and compare the results to our procedure. We ran this on the synthetic data described above. The resulting accuracy loss of each feature was close to the same on the experiments where the model was retrained as it was on the experiments based only on obscuring the test data (see Fig. 1), thus validating the ability to obscure directly on the test data without retraining the model.

Despite the similarity in the final scores, when auditing test data and considering a fixed black-box model the audit reveals some interesting differences between the retrained model that has a chance to adapt to the obscured data and the fixed model. In Fig. 1, the left column shows audits on a decision tree model with the retrained audit on the top and the fixed audit on the bottom. The model considered for the fixed audit has a single node that splits on the Feature A; thus, Feature A is separated from proxy Features B and C for intermediate obscurity values as obscuring Features B and C don't immediately impact the data relative to the chosen split point. The retrained model shows Features A, B and C, tracking closely together as the model adapts to take advantage of the remaining signal.

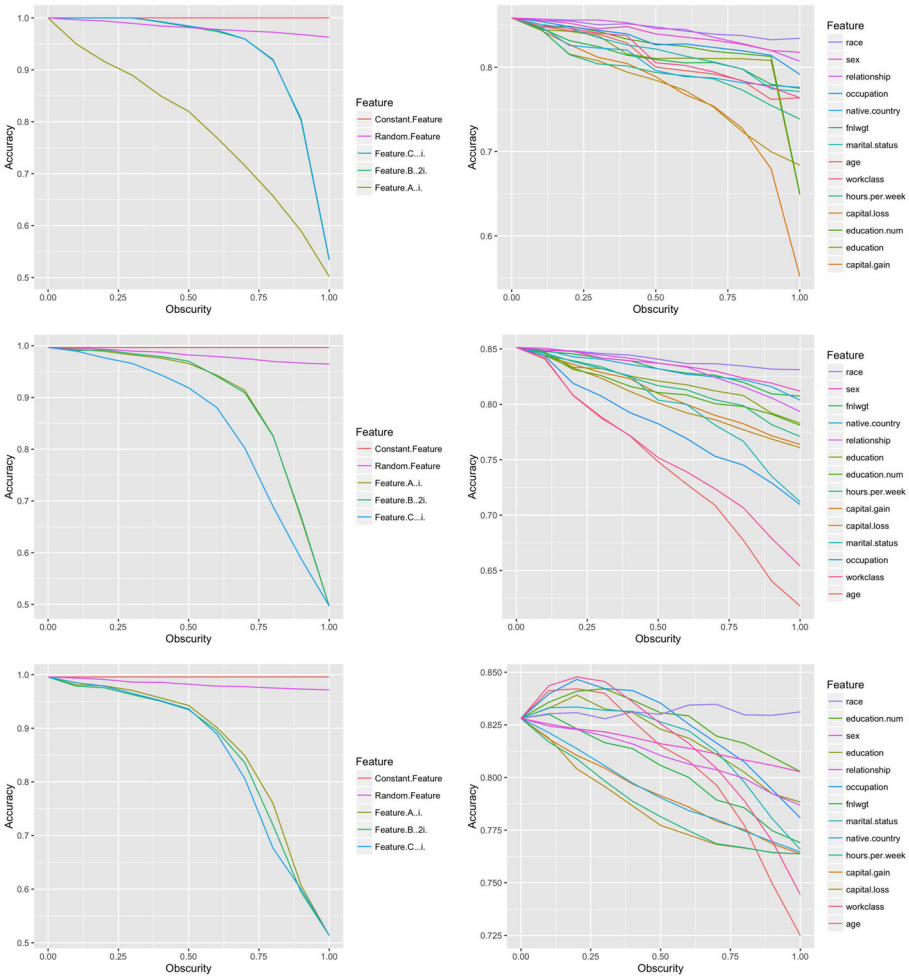


Fig. 2 Obscurity versus accuracy plots for the synthetic data and Adult Income data on each model considered. First column: synthetic data. Second column: adult income data set. First row: C4.5 decision trees. Second row: SVMs. Third column: FNNs

5.2 Black-box feature auditing

We now assess the performance of our GFA method. We trained each model on each of the four data sets. We then ran GFA using the test data for each data set. As we noted in Sect. 4, we progressively increase the degree to which we obscure a data set by removing a variable. Specifically, we used partial obscuring values at 0.1 intervals between 0 (no removal) and 1 (full removal) giving us 11 total partially obscured data sets to consider the accuracy change for. Figures 2 and 3 show the resulting GFA plots.

Synthetic data Beginning with the synthetic data under any of the models, we see that removing any one of the three main Features (*A*, *B* and *C*) that encode the outcome class causes the

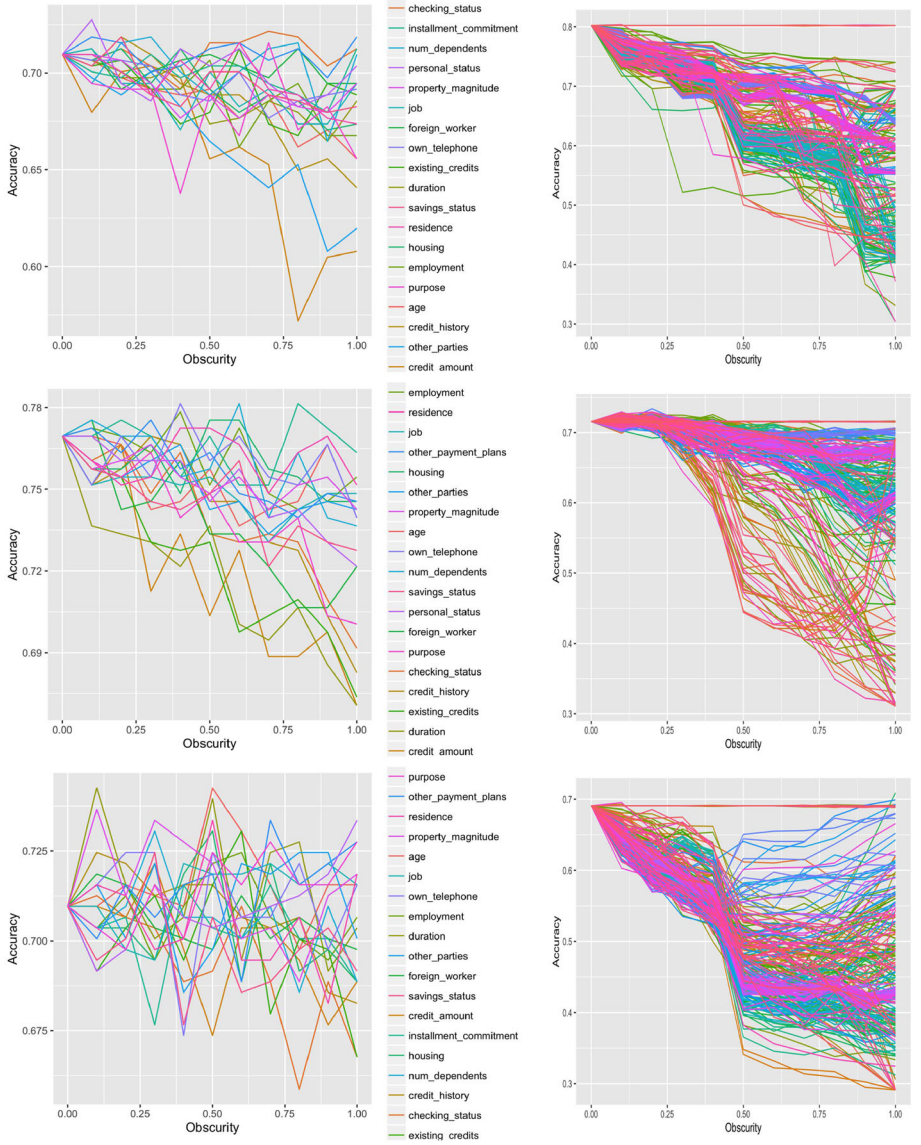


Fig. 3 Obscurity versus accuracy plots for the German Credit data and Dark Reaction data on each model considered. First column: German Credit data. Second column: Dark Reaction data. First row: C4.5 decision trees. Second row: SVMs. Third row: FNNs. Dark Reaction data are shown without a feature legend due to the large number of features

model to degrade to 50% accuracy as our approach would predict. Removing the constant feature has no effect on the model's accuracy, also as expected. The removal of the random feature causing the model to lose a small amount of accuracy may initially seem surprising; however, this is also as expected since the random feature also individually identifies each row and so could be used to accurately train an overfitted model.

Adult Income data On the Adult Income data set, we see that the ranking changes depending on the model. Recall that the more "important," highly ranked features are those that cause the accuracy to drop the most, i.e., are toward the bottom of the charts. While `race` is found to have only a small influence on all the models, the removal of `age` has a large impact on the SVM and FNN models, but is much less important to the decision tree. On the FNN model gradient auditing plot we also see a set of features which when partially removed actually *increased* the accuracy of the model. In a model that was not optimal to begin with, partially obscuring a feature may in effect reduce the noise of the feature and allow the model to perform better.

German Credit data The results on the German Credit data exhibit an arbitrary or noisy ordering. We hypothesize that poor models are likely to produce such auditing results. There are two interrelated reasons why this may be the case. First, since the audit assesses the importance of a feature using the accuracy of the model, if the model is poor, the resolution of the audit is degraded. Second, if the model is poor partially due to overfitting, obscuring features could cause spurious and arbitrary responses. In these contexts, it makes more sense to consider the change in accuracy under a consistency measure. We explore this further in Sect. 5.3.

Dark Reaction data The Dark Reaction data show different top ranked features for the three models, though all three rankings include the minimum Pauling electronegativity and the maximum Pearson electronegativity in the top ranked cluster of features. These values are calculated for the inorganic components of the reaction and have been shown to be important for distinguishing between chemical systems in this data set [24], which this audit confirms. Features indicating the presence of elements and amounts of metal elements with specific valence counts similarly allow the models to classify chemical systems. The SVM and FNN top features include atomic properties of the inorganics that are related to the electronegativity of an element, so these proxies are correctly also highly ranked. The top ranked decision tree features additionally include the average molecular polarizability for the organic components, which was previously hypothesized as important to synthesis of templated vanadium selenites explored via this data set [24]. For all three models, the lowest ranked descriptors are constants scored correctly as having no influence to the model.

Running time Running times for these experiments, including time to train the model, time to do all partially obscured audits and time to write the partially obscured data sets to disk, varied from 13 s on the synthetic data set with the C4.5 decision tree model to just over 3 h for the Dark Reaction data set with the FNN. Since one-time audits are not highly time-sensitive tasks, and the code was unoptimized, we present these times largely as evidence of the feasibility of this method.

5.3 Auditing for consistency

The results in Figs. 2 and 3 allow us to both create a ranking of influence as well as evaluate absolute accuracy of the model on obscured data. However, the German Credit data set yields very noisy results. To address this, we propose using *model consistency*: we replace the original labels with those predicted by the model on unobscured data and then calculate accuracy as we progressively obscure data with respect to these labels. The unobscured data will thus always have a consistency of 100%. The new gradient measured will be the

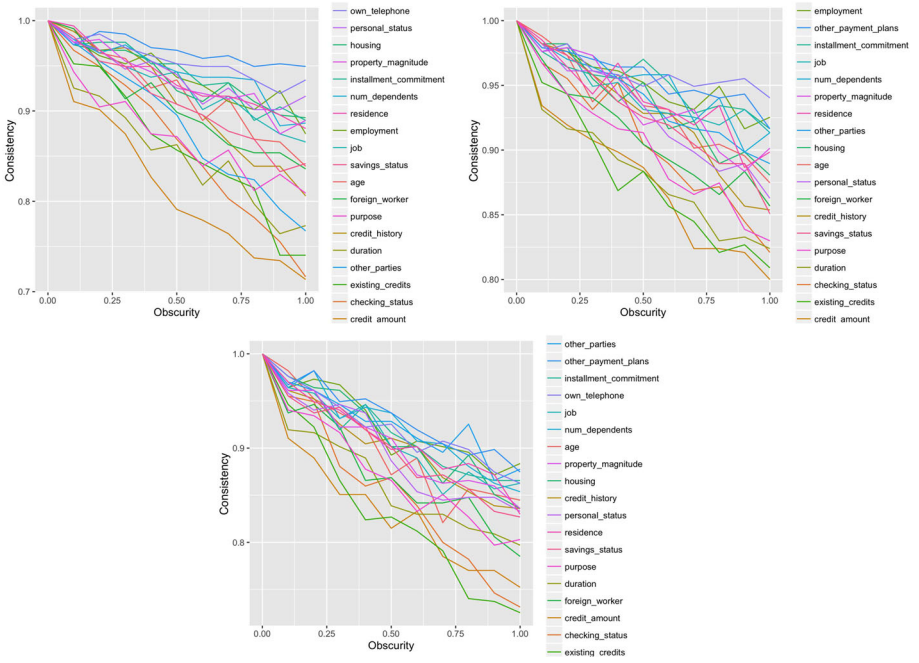


Fig. 4 Obscurity versus consistency plots for the German Credit data. First row: decision tree model (left) and SVM (right). Second row: FNN

difference between the 100% consistency at obscurity of 0 and the degraded consistency when fully obscured.

As shown in Fig. 4, under the consistency measure the accuracy of the German Credit model degrades smoothly so that a ranking can be extracted. The slight noise remaining is likely due to the final step of the categorical obscuring algorithm that redistributes "lost observations" randomly. The resulting ranking is fairly consistent across models, with `credit amount`, `checking status` and `existing credits` ranked as the top three features in all models.

Similar to the German Credit data, the FNN model on the Adult Income data set was not an optimal model. This means that in the accuracy-based plots in Figs. 2 and 3 obscuring the features at first leads, counterintuitively, to an increase in accuracy. In the consistency graph for the FNN model on the Adult Income data (see Fig. 5), we see that while the ranking derived from accuracy closely resembles the consistency ranking, a cluster of features (`native.country`, `capital.loss`, `capital.gain` and `hours.per.week`) had been ranked above `occupation` and `marital.status` and the consistency ranking moves that cluster down in rank.

5.4 Evaluating with respect to a direct influence audit

In order to determine whether GFA is correctly determining the indirect influence of a model, we will first develop a simple method of detecting *direct* influence and then examine the outlying attributes for which our indirect audit differs.

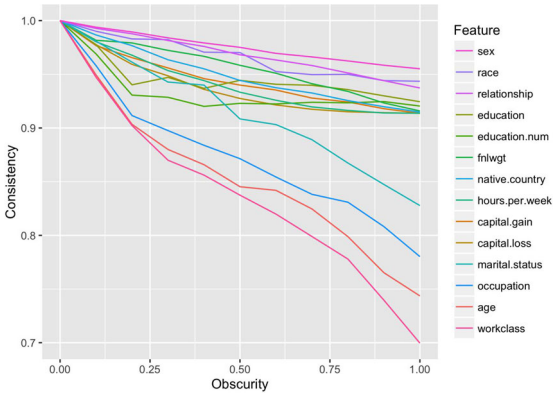


Fig. 5 Obscurity versus consistency for Adult Income data modeled by an FNN

Direct influence audit The first step of the direct influence audit method we will use is the creation of an interpretable model of the model. By a "model of a model," we mean that we should (1) train the model f on training data (X, Y), (2) determine new labels $\hat{Y}$ from the predicted outcomes $f(X)$ and (3) overfit an interpretable model $I(f)$ to these predicted labels (as done in [4]). (This idea is similar to model compression, but without the need to find new test data [6].) Assuming that the model resulting from this procedure has high accuracy on $\hat{Y}$, we now have an interpretable model of our model.

For the SVM and decision tree models trained on the each of the four data sets, we trained an unpruned C4.5 decision tree model of the model. With these interpretable models of a model, unfortunately a manual comparison to the feature ranking is still impossible due to the size of the resulting trees. We create feature importance scores by calculating the probability that each feature appeared on the path from the root to the leaf node containing an item from the training set. This ranking is weighted by the number of items at each leaf node. Any feature appearing at the root node, for example, will have a probability of 1 and come first in the derived ranking. This gives us the direct influence audit results we will compare to.

Synthetic data Beginning with the simple decision tree model for the synthetic data, looking at the decision tree reveals that only Feature A is used explicitly—the decision tree has a single split node. When we create a decision tree model of this model, to confirm the model of a model technique, the result is exactly the same decision tree. Creating a decision tree model of the SVM model, we find again that there is a single node splitting on Feature A. Both models of models have 100% accuracy on the training set (that they were purposefully overfit to). The probability ranking of all of these models for the synthetic data contains Feature A first with a probability of 1 and all remaining features tied after it with a probability of 0 (since only Feature A appears in the decision tree). Since the probability ranking is a direct audit, it is not surprising that it does not score proxy variables B and C highly.

Comparison to a direct influence audit To evaluate the model of a model probability ranking in comparison with the GFA accuracy ranking, we compare the feature rankings on three data sets (Adult, German and Dark Reactions). Specifically, we collect the three generated rankings into one vector for each of the feature ranking procedures and run a Spearman rank correlation statistical test. We find a combined sample rank correlation of 0.263 (a 95%

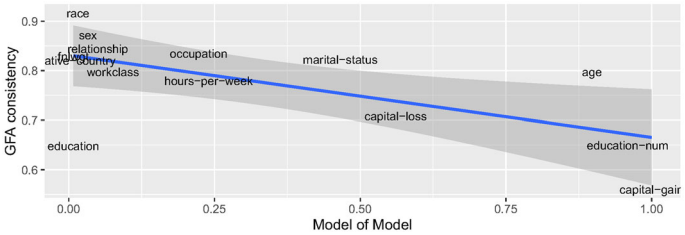


Fig. 6 Model of a model decision tree probability rankings versus GFA consistency scores shown with a linear regression and 95% confidence interval. The outlying features contain proxy information in the data set

Table 1 Synthetic data comparison between Henelius et al. and GFA

	C4.5 Decision tree		SVM	
Feature	Henelius et al.	GFA	Henelius et al.	GFA
A	0.50	0.50	0.87	0.50
B	1.0	0.53	0.87	0.50
C	1.0	0.53	0.88	0.50
Random	1.0	0.96	0.99	0.97
Constant	1.0	1.0	1.0	1.0

All models achieve 100% accuracy, so GFA consistency and accuracy are the same

bootstrap confidence interval of [0.128, 0.387] over 10,000 bootstrap samples), and find also that we can reject the null hypothesis (of no correlation), with $p < 0.002$. When we compare the GFA ranking based on model *consistency* (cf. Sect. 5.3), the results are similar to the ones based on model accuracy. This provides evidence that our feature auditing procedure closely matches a direct influence audit overall.

To consider the cases where these rankings don't match, we look at Adult Income under the C4.5 decision tree model. As shown in Fig. 6, linear regression confirms that most features have similar scores, but there are a few important outliers: `marital-status`, `education`, `race`, `age` and `capital gain`. We hypothesize that the information in these features can be reconstructed from the remaining attributes, and so they are scored differently under an indirect influence audit. We explore this hypothesis next.

5.5 Comparison to previous work

Henelius et al. [16] provide a different technique to solve this black-box feature auditing problem. They focus on determining not only the influence scores associated with each feature, but also groupings of features that are more influential as a group than they are individually (i.e., have mutual influence on a model's outcomes) and use the consistency measure as the score. The results of running their algorithm[8] on the synthetic data we considered here is shown in Table 1.

These scores on the synthetic data set illuminate the key difference between the Henelius et al. work and GFA: while Henelius et al. focus on auditing for *direct influence* on a model, GFA includes both direct and *indirect influence* through proxy variables. For example, the C4.5 decision tree created on the synthetic data is a very simple tree containing one node that splits on the value of Feature A. Henelius et al. correctly show that A is the only feature directly used by the model. However, GFA additionally shows that Features B and C are

[8] Available at: https://bitbucket.org/aheneliu/goldeneye/.

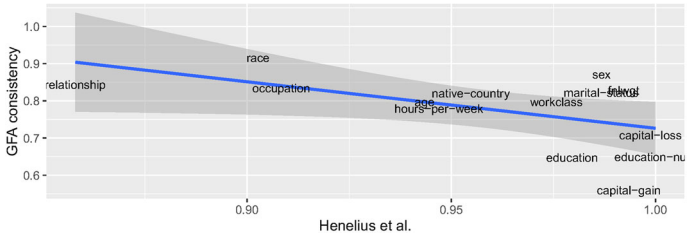


Fig. 7 Henelius et al. influence scores versus GFA consistency scores shown with a linear regression and 95% confidence interval. The outlying features contain proxy information in the data set

Table 2 Adult Income data comparison between Henelius et al. and GFA consistency and accuracy scores for a C4.5 decision tree model

Feature	C4.5 audit scores			Feature predictability	
	Henelius et al.	GFA cons.	GFA acc.	REPTree	C4.5 or M5
Capital gain	0.94	0.56	0.55	0.16	0.21
Education	0.98	0.65	0.65	1.0	1.0
Education-num	0.92	0.65	0.65	1.0	1.0
Capital loss	0.98	0.71	0.68	0.09	0.16
Hrs per week	0.96	0.78	0.74	0.44	0.49
Age	0.94	0.80	0.76	0.65	0.68
Workclass	0.98	0.80	0.76	0.11	0.16
Fnlwgt	0.99	0.83	0.77	0.15	0.22
Marital-status	0.87	0.82	0.77	0.74	0.76
Native country	1.0	0.82	0.78	0.22	0.28
Occupation	0.90	0.84	0.79	0.21	0.17
Relationship	0.99	0.84	0.81	0.69	0.70
Sex	1.0	0.87	0.82	0.62	0.62
Race	1.0	0.92	0.83	0.25	0.23

Feature predictability scores are correlation coefficient or kappa statistic (for numerical or categorical features, respectively) when predicting that feature from the remaining features using two tree-based models

proxies for A (recall that B is defined as two times Feature A and C is defined as negative one times Feature A).

For a real-world comparison to Henelius et al., we consider the Adult data set under a C4.5 decision tree model. The scores and rankings generated by Henelius et al. do not match those generated by GFA. Figure 7 shows that features `marital-status`, `fnlwgt`, `sex`, `education`, `education-num` and `capital gain` are outliers. In order to determine whether this is due to the presence of proxy variables, or variables that are more complexly encoded in the remaining attributes by the decision tree, we then used two tree-based models to predict each feature from the remaining features (see Table 2).[9] Models were built on the test set (using a $\frac{2}{3} : \frac{1}{3}$ training test split) in order to replicate the data used for the audit. Reported predictability scores are the correlation coefficient for numerical features and the kappa statistic for categorical features.

[9] Weka's REPTree, J48 and M5P models were used for this analysis with the default model-building parameters. J48 was used to predict categorical features and M5P was used for numerical features. REPTree can handle both categorical and numerical features.

Looking at the resulting feature predictability scores, we see that `education` and `education-num` are both perfectly reconstructable from the remaining attributes. This is not surprising since `education-num` is a numerical representation of `education` and thus a perfect proxy. The GFA consistency and accuracy measures both have `education` and `education-num` as tied for an importance score of 0.65, thus confirming that GFA handles indirect influence, while Henelius et al. has `education` with a score of 0.98 while `education-num` scores 0.92.

The features `marital-status` and `relationship` are both highly, but not exactly, predictable from the remaining attributes. This is likely because these are close, but not exact, proxies for each other. For example, the relationship status "Unmarried" may mean a marital status of "Divorced," "Never married," or "Widowed." The GFA scores for `marital-status` and `relationship` show they are of similar importance to the model, with consistency scores of 0.82 and 0.84, respectively (ranked 8.5 and 11.5 in importance), and accuracy scores of 0.77 and 0.81 (ranked 8.5 and 12). The Henelius et al. scores are less similar at 0.87 (ranked most important) and 0.99 (ranked 10.5). The GFA closely matching scores shows the procedure accounts for indirect influence of a feature, while Henelius et al. does not.

Recent work by Datta et al. [11] (discussed in more depth in Sect. 2) presents a similar approach, focusing on direct influence, that can additionally identify proxy variables. Identifying the influence of proxy variables using this method proceeds in two steps: first, the proxy variables are identified; then, their individual direct influence in the ranked list is found. Considering the Adult data set, but under a different model, they find that `marital-status` and `relationship` are both proxies for `sex`. Their ranking finds that `marital-status` is ranked first for influence, `relationship` third and `sex` ninth. Additionally, their influence score for `relationship` is under half of that for `marital-status`, and the influence score for `sex` is under half of that for `relationship`. Thus, similar to Henelius et al., the Datta et al. two step procedure does not account for the shared indirect influence of `marital-status` and `relationship` on the outcome.

5.6 Comparison to feature selection

Finally, we examine the relation between our auditing procedure and feature selection methods. While GFA is fundamentally different from feature selection since (a) features may not be removed from the model and (b) the model may not be retrained, due to their similar ranked feature results, we compare the resulting GFA ranking to a feature selection generated ranking. Feature selection was applied to the synthetic, Adult Income and German Credit data sets. It was performed using a wrapper method (around both C4.5 decision trees and SVMs) and a greedy forward search through attribute subsets to generate a ranking.[10] For both the Adult Income and German Credit data sets, feature selection created identical rankings for both decision trees and SVMs.

Spearman's rank correlation is a nonparametric test of the relationship between two rankings, in this case the rank orderings generated by feature selection and by GFA. The synthetic data had a strong correlation between these rankings, with Feature *A* ranked first by all methods, though given the small feature size this is not statistically significant. The Adult Income and German Credit data set ranking comparison correlations were both weak. This was not

[10] Feature selection was implemented in Weka version 3.6.13 using WrapperSubsetEval and Greedy StepWise on J48 and SMO models. Default options were used, save for the generation of a complete ranking for all features.

improved when using the consistency feature ranking instead and in the case of the C4.5 decision tree was an order of magnitude weaker.

The weak correlation is not surprising since the two methods ask philosophically different questions. We are interested in the importance of a feature to a specific instance of a model, while feature selection considers importance with respect to an as-yet uninstantiated model. In addition, feature selection gives an indication of the direct influence of a feature, while GFA is focused on indirect influence.

6 Analysis of COMPAS recidivism prediction software

Increasingly, machine learning algorithms are being used to predict whether a defendant in a criminal case will re-offend. These models, such as one developed by Northpoint called COMPAS (Correctional Offender Management Profiling for Alternative Sanctions), are then used by judges to help inform sentencing decisions. Recent work by ProPublica [3] collected an extensive data set containing criminal records of 7215 people from Broward County, Florida, including demographic information such as their age, sex and race, juvenile record information such as the counts of felonies and misdemeanors, information associated with the charge they were arrested for, and the COMPAS score label (low, medium or high) at booking for each of these people. The ProPublica analysis, comparing these scores against the records of whether the defendants actually recidivated over the following two years, found that the scores were biased against African-Americans in the sense that, of the people who were misclassified, African-Americans were more likely to receive a misclassification indicating that they were high risk and not go on to re-offend, while white defendants were more likely to receive a misclassification indicating that they were low risk and then go on to recommit. This work has spurred an interesting discussion about definitions of fairness and the impossibility of achieving all types of fairness when the base rates of arrest (how recidivism is measured) are different for different demographic groups [9,19]. To the best of our knowledge, COMPAS is the only recidivism model in active use to have received any public scrutiny. Given its potential impact in human lives, it is an important case study.

These discussions of fairness center around different proportions of outcomes across positive and negative classes for different demographic groups. Here, we are interested in auditing the COMPAS algorithm to start to answer the question of *how* the COMPAS model made these decisions, with the goal of allowing a person charged with auditing the algorithm to understand the potential for discrimination in the model. Our goal will be to demonstrate the techniques by using this interesting example model and data set, not to come to a definitive understanding of how the COMPAS algorithm works. This limitation is due to the lack of access to run the COMPAS algorithm, or even to access the specific features that COMPAS used. Our auditing methods described so far assume that we have access to both the inputs and the outputs of an algorithm and the ability to run the algorithm ourselves on purposefully chosen inputs. Instead, here, we have access to the COMPAS scores (outputs) and a modified set of inputs. We don't have the ability to rerun the algorithm. Under these constraints, we will aim to give a demonstration of how these techniques might be used in practice.

Data and model preparation ProPublica's data include records for 7215 individuals, but they suggest[11] filtering the data set down to 6172 individuals to include only those people who have complete data—they contain a COMPAS score, and it has been at least two years

[11] The ProPublica methodology can be found here: https://github.com/propublica/compas-analysis.

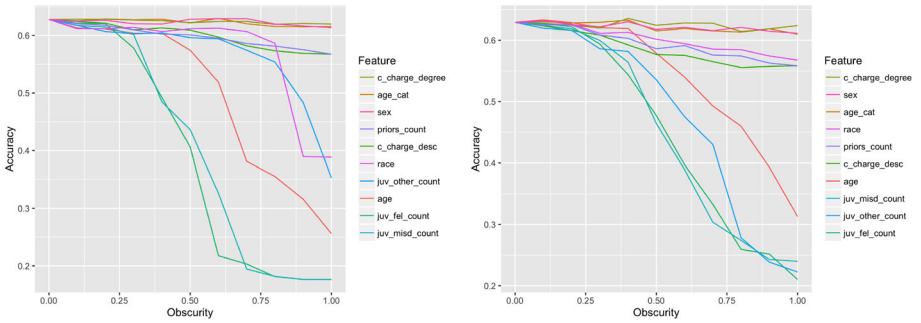


Fig. 8 Obscurity versus accuracy charts for the ProPublica data set when the outcome predicted is the COMPAS score label (low, medium or high). The model used to predict this COMPAS score is a C4.5 decision tree (left) or an SVM (right)

since that score was calculated (so the two year recidivism outcome is valid). Individuals with traffic offenses are removed, as are individuals who don't have a clear COMPAS score assessment associated with the arrest record. Additionally, we exclude date features, arbitrary strings and other features such as name or case number that are specific to each individual. We keep only the following 10 features in our training set: the juvenile felony count, juvenile misdemeanor count, other juvenile records count, race, sex, a count of the prior records, the charge description for this arrest, the charge degree (felony or misdemeanor) for this arrest, age and a categorical version of age (less than 25, 25–45 and greater than 45). Our prediction problem focuses on replicating the COMPAS score—we predict the low-/medium-/high-score label for the regular recidivism problem (and not for violent recidivism). We randomly choose two-thirds of the data for training and one-third for testing.

6.1 Auditing COMPAS for indirect influence

We begin our analysis of the COMPAS model by using our developed techniques to perform an indirect influence audit using the ProPublica developed input and the COMPAS low-, medium- and high-score labels as the predicted output. We use both decision trees and SVMs to model the COMPAS model, and perform the audits with respect to these models of the model. The audit results are shown in Fig. 8.

Interestingly, the decision tree and SVM audits of the COMPAS model are different. Neither is able to perfectly predict the score outcomes, likely because we don't have all the features that were used to create these outcomes. Juvenile charges and age are shown to have a large impact under both of these models. Charge degree (felony or misdemeanor), age category (below 25, 25–45 and above 45) and sex have very little influence under either model. Race, on the other hand, is found to be important under both models of COMPAS but found to have more influence for the decision tree (0.24 out of an accuracy of 0.63) than the SVM (0.06 out of 0.63). Ideally, we'd be able to examine how race indirectly impacts a model of COMPAS more specifically. We will build up to this examination in what follows.

6.2 Interpretable indirect influence

In order to develop an interpretable model that we will use later to examine the indirect influence of race on our model of COMPAS, we use the CN2 rule induction algorithm [10] to develop the full rule set, again using CN2 to predict the COMPAS score labels. Unfortunately, despite the interpretability of these rules, it is very hard to reason about whether there is

indirect influence based on race in this model. In response, we introduce a procedure to create and score additional modified rules with respect to their indirect influence based on race. Our goal with this procedure is to be conservative in our influence scoring. We aim to determine which rules of the original set *do not* have indirect influence based on (say) race and therefore which remaining rules have potential indirect racial impact. An outline of this procedure to audit rules for indirect influence is as follows:

1. Create a rule set modeling the outcomes. The CN2 rule models we train are generated using Orange.[12]
2. Create an obscured version of the training data with respect to the chosen removed class (in this case, race).
3. Convert the original rules into obscured rule sets so that rule variable values are transformed to the new obscured values. Do this for all combinations of features and for all possible obscured values. In what follows, we will use "Feature_A_noFeatureB" to mean a modified version of Feature A that has been obscured with respect to Feature B. For example, "Feature_Age_noRace" denotes the age feature obscured with respect to race.
4. Determine a quality score and an influence score for each rule.
5. Choose some threshold ϵ. Consider all transformed rules that are within ϵ quality of the original rule.
6. Output the rule set consisting of the rules chosen with the lowest potential indirect influence score from the set within ϵ quality of the original rule.
7. Any rule in the output set that contains an unobscured feature has potential indirect racial influence.

Obscuring rule sets In order to create obscured and partially obscured versions of the CN2 rules, we need to map each feature and associated value in the rule to its obscured counterpart. Since we would like to examine the best quality rule with the least potential for indirect influence, we will create each obscured feature independently, so that all combinations of obscured and unobscured features in the rule set are considered. The value of chosen obscured feature value needs to be mapped to the new obscured value, and there are potentially multiple of these, one per value of the removed attribute. For numerical attributes, the boundary value (e.g., 3 if the rule includes Feature$A <= 3$) is mapped to its possible obscured values. We consider all of these values in the possible generated rules.

Scoring rule quality Each rule is scored for quality using the Laplace heuristic: quality $= \frac{\text{correct}+1}{\text{total}+k}$ where total is the number of instances covered by the rule, correct is the number of those that were correctly classified and k is the number of classes in the classification problem.

Scoring indirect rule influence Given a rule with m features f_i, we are interested in being able to compare the indirect influence of this rule based on the removed feature to the original rule. We calculate the indirect rule influence as $\sum_{i=1}^{m} \text{influence}(f_\text{i})$ where $\text{influence}(f_\text{i})$ is the amount of indirect influence with respect to the removed feature as determined using our described auditing method, and $\text{influence}(f_\text{i})$ is defined as 0 when the obscured version of f_i is used. These influence scores will only be used to compare rules that have the same number of features, so by summing the influence scores per feature (instead of, for example, using the maximum) we make sure a rule with multiple potentially influential features is appropriately marked as having higher influence without suffering from normalization issues.

[12] http://orange.biolab.si/toolbox/.

Table 3 Original rules for the CN2 model of the decision tree model on the synthetic data are given first

Original Rules	Quality	
IF Feature_A >= 3001 THEN Outcome = B	1.0	
IF TRUE THEN Outcome = A	1.0	
Possible Rules—Constant removed	Quality	Influence
IF Feature_A >= 3001 THEN Outcome = B	1.0	0.5
IF Feature_A_noConstant_Feature >= 3001 THEN Outcome = B	1.0	0.0
Possible Rules—Feature A removed	Quality	Influence
IF Feature_A >= 3001 THEN Outcome = B	1.0	0.5
IF Feature_A_noFeature_A >= 3593 THEN Outcome = B	0.5	0.0
Possible Rules—Feature B removed	Quality	Influence
IF Feature_A >= 3001 THEN Outcome = B	1.0	0.5
IF Feature_A_noFeature_B >= 3001 THEN Outcome = B	0.56	0.0

These are followed by possible rules and scores to replace the rule `IF Feature_A >= 3001 THEN Outcome = B` when obscured variables with respect to a removed feature are considered. The removed features considered are the constant feature, Feature A and Feature B. Influence scores are calculated with respect to the rule features as described above

6.3 Experiments on synthetic data

In order to validate this new method, we first considered the rules generated for the decision tree and SVM models of our synthetic data set. Recall that Features *A*, *B* and *C* in the synthetic data are perfect proxies for each other, since each can be used by itself to exactly determine the outcome. In addition, the decision tree model of these data includes a single node determining the outcome directly based on Feature *A*. The output from using CN2 to model the decision tree model of the synthetic data also uses only Feature *A* and is given in Table 3.

For the purposes of this smaller synthetic data set, we will avoid choosing a specific value for ϵ and instead look at all possible full and partially obscured rules that could be generated and evaluate them for quality and influence. We obscure the rules with respect to the constant feature, Feature A and Feature B (see Table 3). We expect that the rule quality will not go down when the rules are obscured with respect to the constant feature, while the rule quality will decrease to essentially random when the rules are obscured with respect to both Feature A (demonstrating a removal of direct influence) and Feature B (indirect influence). As expected, we see that we can decrease the potential influence of the constant feature on the rule without changing the quality, while decreasing the potential influence of Features A and B on the rule also decreases the quality. From this, we can see that the rule has indirect influence based on Features A and B that impacts the resulting predictions.

6.4 COMPAS results

We used CN2 to create a model of the low-/medium-/high-score results and found that we were able to model them with an accuracy of 75%. Sixty-nine rules were created and race was explicitly used in 36 of these rules. But assuming that these rules represent the entirety of

discrimination in the rule set is misleading in two ways. First, it may be possible to obscure the rule with respect to race and not have the quality of the rule suffer—in that case, we could consider the original rule nondiscriminatory. Second, there may be other rules that have indirect influence based on race that *should* be marked as discriminatory and wouldn't be by just identifying rules containing race directly. In order to investigate both of these possibilities, we perform the interpretable indirect influence audit described above, using the influence scores as calculated assuming the decision tree model.

Example rule examinations Let us look more closely at how this technique impacts our examination of a single rule:

```
IF priors_count<=3 AND age>=50 AND race !=African-American
 AND
    c_charge_desc=!Grand Theft in the 3rd Degree AND
    c_charge_desc!=Possession of Cocaine
THEN score_text=Low
```

This rule has a quality score of 0.99 when evaluated on the test set, and its influence score is 0.79. The expansion of this rule to consider all possible obscured and unobscured versions of these 5 features along with all possible mappings to the corresponding obscured values generates 1521 versions of this rule. In total, 162 of the resulting rules have 0 influence based on race, i.e., all features considered are the obscured versions of the feature for different possible obscured values. All the rules with 0 racial influence also have a quality of 0.33, or the equivalent of choosing low/medium/high randomly. For all reasonable values of ϵ, i.e., for any values that do not admit a quality of 0.33, the rule with the lowest influence score still retains an explicit dependence on race. For example, with $\epsilon = 0.1$, the rule with quality within 0.1 of the original rule that has the lowest potential influence based on race is:

```
F priors_count-norace<=5 AND age-norace>=47 AND
    race!=African-American AND
    c_charge_desc-norace!=Grand Theft in the 3rd Degree AND
    c_charge_desc-norace!=Felony Battery (Dom Strang)
THEN score_text=Low
```

This rule has a test set quality score of 0.90 and an influence score of 0.24, all of which is due to the direct influence of race on the outcome. This shows us a specific subset of individuals for whom the recidivism predictor acts differently based on race.

But the presence of race in a rule does not always indicate that race influences the outcome. For example, this rule has a quality score of 0.69 and an influence score of 0.39:

```
IF age_cat==Greater than 45 AND priors_count<=15 AND
    race==Caucasian AND
    c_charge_desc!=Grand Theft in the 3rd Degree AND sex!=
Female
THEN score_text=Low
```

One version of this rule with each feature fully obscured is:

```
IF age_cat-norace==Greater than 45 AND priors_count-norace<=5
 AND
    race-norace==African-American AND sex-norace!=Male AND
    c_charge_desc-norace!=Grand Theft in the 3rd Degree
THEN score_text=Low
```

The quality score actually goes up on the test set to 0.82, while the influence score is 0, showing that despite the explicit reference to race in the rule, race does not have a large impact on the part of the model's outcome dictated by this rule.

High influence rules One way to achieve insight into the potential discriminatory harm of the data set is to consider the rule with highest potential influence based on race (influence score 0.73, $\epsilon = 0.01$). That rule is:

```
IF juv_other_count-norace>=1 AND race!=African-American AND
    age_cat!=25-45 AND race!=Hispanic AND race!=Other
THEN score_text=Medium
```

In this rule, the explicit determination based on race is retained for all values of ϵ small enough to avoid a quality score of 0.33 indicating random choice. Here, that means that the determination of a medium COMPAS risk score for White and Asian defendants between 25 and 45 with at least one juvenile infraction is heavily influenced by race (also implying that African-American and Hispanic defendants with similar characteristics are denied a medium risk score).

Analysis of the resulting rule set Considering the rule set as a whole, we are especially interested in determining, for various values of ϵ, how many rules still retain influence based on race. Recall that 36 of the 69 rules original depended explicitly on race. For $\epsilon = 0.01$, this decreases to 18 rules based explicitly on race, showing that the other 18 rules were not as heavily dependent on racial influence as might have been assumed from the explicit reference to race in the rule. For larger $\epsilon = 0.1$, there are still 17 rules based explicitly on race, indicating that these remaining rules do influence the outcome based on race. Similarly, for $\epsilon = 0.01$ there are 50 of the 69 rules that retain some influence based on race, while at $\epsilon = 0.1$ this decreases to 39 rules with nonzero influence scores.

Limitations, threats to validity In this section, as well as in Sect. 5, we have used the technique of building a *model of a model* to assess the behavior of data-driven automated systems. Although Sect. 5.3 indicates that such models of models match existing models relatively well, there are limitations to the method which we want to point out. First, it is conceivable that the actual model used did not employ the features highlighted by the model of a model technique. Still, this threat to validity is mitigated by our method's ability to pinpoint indirect influence, since indirect influence happens when the model uses *any* of the features that are correlated with one another. This points to a second threat to validity, which is indirect influence is dependent on a distribution of training and test examples. If the data distribution substantially changes, this will have an effect on our audit. However, in the case of such changes, the entire decision process is likely to have to change as well. We envision our model assessment method as a tool to suggest further investigations, and not as a single source of determination of influence or discrimination, especially if the accuracy of the model of model is substantially lower than that of the original model.

7 Discussion

Feature influence is a function of the interaction between model and data. A feature may be informative but not used by the classifier, or conversely might be a proxy and still useful. Thus, influence computation must exploit the interaction between model and data carefully.

If the obscured and unobscured data sets are similar, then the classifier can't have found useful signal and the classifier's outputs won't change much under our audit. If there *were* significant differences and the classifier used these differences in its model, then gradient feature auditing will show a change in the classifier behavior, as desired. Finally, if there were differences between attribute subgroups but those differences are irrelevant for the classifier, then gradient feature auditing will not show a large change in classifier behavior. This "sensitivity to irrelevance" is an important feature of a good auditing procedure.

It remains a challenge to effectively compare different approaches for auditing models since, as we have seen, different approaches can have points of agreement and disagreement. Our obscuring procedure prefers to use a computational metaphor—predictive power—rather than a statistical metaphor such as hypothesis testing, but it seems likely that there are ways to relate these notions. Doing so would provide a combined mathematical and computational framework for evaluating black-box models and might help unify the different existing approaches to performing audits.

8 Related work

In addition to early work by Breiman [5] and the recent works by Henelius et al. [16] and Datta et al. [11], a number of other works have looked at black-box auditing, primarily for direct influence [12,28,29,31]. There are potential connections to privacy-preserving data mining [2]; however, the trust model is different: in privacy-preserving data mining, we do not trust the user of the results of classification with sensitive information from the input. In our setting, the "sensitive" information must be hidden from the classifier itself. Another related framework is that of *leakage* in data mining [18], which investigates whether the methodology of the mining process is allowing information to leak from test data into the model. One might imagine our obscuring process as a way to prevent this: however, it might impair the efficacy of the classifier.

Another related topic is *feature selection*, as we discussed in Sect. 5.6. From a technical standpoint (see [8]), the *wrapper* approach to feature selection (in which features are evaluated based on the quality of the resulting prediction) is most related to our work. One such method is stepwise linear regression, in which features are removed from input for a generalized linear model based on their degree of influence on the model to make accurate predictions, measured using a correlation coefficient.

Model *interpretability* focuses on creating models that are sparse, meaningful and intuitive, and thus, human-understandable by experts in the field the data are derived from [30]. The classic example of such models are decision trees [23], while recently supersparse linear integer models (SLIMs) have been developed as an alternative interpretable model for classification [30]. New work by Ribeiro et al. [25] trains a shadow interpretable model to match the results of a given classifier. For neural networks, various approaches based on visualizing the behavior of neurons and inputs have also been studied [17,20,32].

Finally, we note that a compelling application for black-box audits of indirect influence includes considerations of algorithmic fairness [26]. Understanding the influence of variables can help with such a determination.

9 Conclusion

In this paper, we have shown that it is possible to audit models even in the presence of proxy variables and that it is possible to detect such indirect influence in a variety of settings. The

method we introduce depends critically on *obscuring* features and measuring the sensitivity of models to such changes. Because the obscuring process is dependent on the distribution of data, this presents important questions for future research. Specifically, we are interested in the interplay between training and test distributions, the models under scrutiny and definitions of influence, discrimination or lack thereof. As different techniques to assess predictive models make fundamentally different assumptions, we see the investigation of precise definitions as enabling novel and improved methods for auditing black models.

References

1. Adler P, Falk C, Friedler SA, Rybeck G, Scheidegger C, Smith B, Venkatasubramanian S (2016) Auditing black-box models for indirect influence, In: Proceedings of the IEEE international conference on data mining (ICDM)
2. Agrawal R, Srikant R (2000) Privacy-preserving data mining, In: ACM Sigmod Record, vol 29. ACM, pp. 439–450
3. Angwin J, Larson J, Mattu S, Kirchner L (2016) Machine bias, ProPublica
4. Barakat N, Diederich J (2004) Learning-based rule-extraction from support vector machines. In: Proceedings of the 14th international conference on computer theory and applications
5. Breiman L (2001) Random forests. Mach Learn 45(1):5–32
6. Bucilua C, Caruana R, Niculescu-Mizil A (2006) Model compression, In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 535–541
7. Casella G, Berger RL (2001) Statistical inference, 2nd edn. Cengage Learning, Boston
8. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40:16–28
9. Chouldechova A (2016) Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. In: Presented at the workshop on fairness, accountability, and transparency in machine learning (FATML)
10. Clark P, Niblett T (1989) The cn2 induction algorithm. Mach Learn 3(4):261–283
11. Datta A, Sen S, Zick Y (2016) Algorithmic transparency via quantitative input influence: theory and experiments with learning systems. In: Proceedings of 37th IEEE symposium on security and privacy
12. Duivesteijn W, Thaele J (2014) Understanding where your classifier does (not) work—the SCaPE model class for EMM, In: International conference on data mining (ICDM), pp 809–814
13. Feldman M, Friedler SA, Moeller J, Scheidegger C, Venkatasubramanian S (2015) Certifying and removing disparate impact. In: Proceedings of the 21st ACM KDD, pp 259–268
14. Freedman D, Diaconis P (1981) On the histogram as a density estimator: L 2 theory. Probab Theory Relat Fields 57(4):453–476
15. Hastie T, Tibshirani R (1998) Classification by pairwise coupling. In: Jordan MI, Kearns MJ, Solla SA (eds) Advances in neural information processing systems, vol 10. MIT Press, Cambridge
16. Henelius A, Puolamäki K, Boström H, Asker L, Papapetrou P (2014) A peek into the black box: exploring classifiers by randomization. Data Min Knowl Disc 28:1503–1529
17. Kabra M, Robie A, Branson K (2015) Understanding classifier errors by examining influential neighbors. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3917–3925
18. Kaufman S, Rosset S, Perlich C, Stitelman O (2012) Leakage in data mining: Formulation, detection, and avoidance. ACM Trans Knowl Discov Data (TKDD) 6(4):15
19. Kleinberg J, Mullainathan S, Raghavan M (2017) Inherent trade-offs in the fair determination of risk scores, In: Proceedings of innovations in theoretical computer science (ITCS)
20. Le QV, Ranzato M, Monga R, Devin M, Chen K, Corrado GS, Dean J, Ng AY (2011) Building high-level features using large scale unsupervised learning. In: Proceedings of the ICML
21. Massey DS, Denton N (1993) American apartheid: segregation and the making of the underclass. Harvard University Press, Cambridge
22. Motwani R, Raghavan P (1995) Randomized Algorithms. Cambridge University Press, Cambridge
23. Quinlan R (1993) C4.5: programs for machine learning. Morgan Kaufmann Publishers, San Mateo
24. Raccuglia P, Elbert KC, Adler PDF, Falk C, Wenny MB, Mollo A, Zeller M, Friedler SA, Schrier J, Norquist AJ (2016) Machine-learning-assisted materials discovery using failed experiments. Nature 533:73–76
25. Ribeiro MT, Singh S, Guestrin C (2016) Why Should I Trust You?: Explaining the Predictions of Any Classifier. In: Proceedings of the ACM KDD
26. Romei A, Ruggieri S (2014) A multidisciplinary survey on discrimination analysis. Knowl Eng Rev 29:582–638

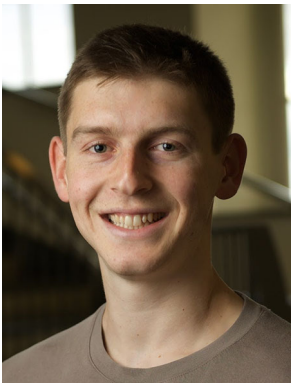
27. Rubner Y, Tomasi C, Guibas LJ (1998) A metric for distributions with applications to image databases. In: 6th International conference on computer vision 1998. IEEE, pp 59–66
28. Strobl C, Boulesteix A-L, Kneib T, Augustin T, Zeileis A (2008) Conditional variable importance for random forests. BMC Bioinf 9(1):1
29. Strobl C, Boulesteix A-L, Zeileis A, Hothorn T (2007) Bias in random forest variable importance measures: illustrations, sources and a solution. BMC Bioinf 8(1):1
30. Ustun B, Traca S, Rudin C (2014) Supersparse linear integer models for interpretable classification. Technical report 1306.6677, arXiv
31. Zacarias OP, Bostrom H (2013) Comparing support vector regression and random forests for predicting malaria incidence in Mozambique. In: International conference on advances in ICT for emerging regions (ICTer), 2013. IEEE, pp 217–221
32. Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: Computer vision—ECCV 2014. Springer, pp 818–833

Philip Adler is presently a software developer in cybersecurity. He received a Ph.D. in Cheminformatics from Southampton University in 2017, having written up while working as a postdoctoral researcher at Haverford College in Pennsylvania during 2015–2016. He maintains an active hand in academic research in his spare time, primarily by contributing to open source projects and helping academic contacts with thorny data and software issues.

Casey Falk currently works as a software engineer at Amazon in Seattle, WA. Previously, he worked at MITRE in the domain of artificial intelligence. He received his Bachelors in Computer Science from Haverford College in 2016. Advised by Dr. Friedler, his research included topics such as machine learning, recommender systems and model audibility.

Sorelle A. Friedler is an Assistant Professor of Computer Science at Haverford College and an Affiliate at the Data & Society Research Institute. Her work focuses on developing algorithms to enhance the fairness, accountability and transparency of machine learning techniques, and she has been the program committee co-chair and an organizer of FAT* and FAT/ML. She received a Ph.D. in Computer Science from the University of Maryland, College Park, and a B.A. in Computer Science from Swarthmore College.

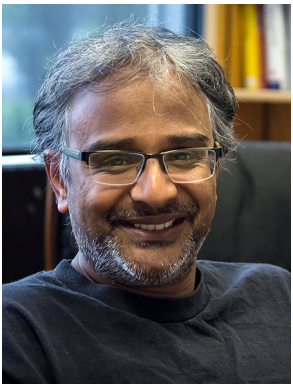
Tionney Nix is currently a software engineer at Google. She received her Bachelor of Science in Computer Science at Haverford College. Advised by Dr. Friedler, she authored a senior thesis titled "A Rule Learning Approach to Discovering Contexts of Discrimination," which focused on using a combination of the Gradient Feature Auditing algorithm and the CN2 rule learning algorithm as an interpretable method for learning which groups in data might be discriminated against.



Gabriel Rybeck is a software engineer and data scientist at Booz Allen Hamilton in Washington, D.C. He received his Bachelors in Computer Science and Economics from Haverford College in 2016. Advised by Dr. Friedler, he authored a senior thesis entitled "Indirect Discrimination in the Age of Big Data," which explores the unintended consequences of using black-box machine learning algorithms to make important decisions. His current work focuses on enabling big data analytic capabilities for the intelligence community, with an emphasis on interpretable machine learning algorithms.

Carlos Scheidegger is an Assistant Professor in the Department of Computer Science at the University of Arizona. His research interests are in large-scale data analysis, information visualization and, more broadly, what happens "when people meet data." He received his Ph.D. from University of Utah and a B.Sc. in Computer Science from the Federal University of Rio Grande do Sul.

Brandon Smith leads the data science and product development teams at PrecisionGx in Philadelphia, PA. He received his Bachelors in Computer Science from Haverford College in 2016 and Masters in Robotics from the University of Pennsylvania in 2017. Advised by Dr. Friedler, he authored a senior thesis entitled "Auditing Deep Neural Networks to Understand Recidivism Predictions," which examined techniques for identifying biases in black-box machine learning models. His current work focuses on utilizing health insurance claims data to build machine learning models that drive smarter decision making within healthcare organizations.



Suresh Venkatasubramanian is a Professor in the School of Computing at the University of Utah. His research interests lie in the area of fairness, accountability and transparency in machine learning. He received his Ph.D. from Stanford University and a B.Tech in Computer Science and Engineering from the Indian Institute of Technology, Kanpur.