Auditing Black-box Models by Obscuring Features*

Philip Adler, Haverford College, padler1@haverford.edu
Casey Falk, Haverford College, cfalk@haverford.edu
Sorelle A. Friedler, Haverford College, sorelle@cs.haverford.edu
Gabriel Rybeck, Haverford College, grybeck@haverford.edu
Carlos Scheidegger, University of Arizona, cscheid@cscheid.net
Brandon Smith, Haverford College, bpsmith@haverford.edu
Suresh Venkatasubramanian, University of Utah, suresh@cs.utah.edu

Abstract

Data-trained predictive models are widely used to assist in decision making. But they are used as *black boxes* that output a prediction or score. It is therefore hard to acquire a deeper understanding of model behavior: and in particular how different attributes influence the model prediction. This is very important when trying to interpet the behavior of complex models, or ensure that certain problematic attributes (like race or gender) are *not* unduly influencing decisions.

In this paper, we present a technique for *auditing* black-box models: we can study the extent to which existing models take advantage of particular features in the dataset without knowing how the models work. We show how a class of techniques originally developed for the detection and repair of disparate impact in classification models can be used to study the sensitivity of any model with respect to any feature subsets.

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 like feature selection. We present experimental evidence for the effectiveness of our procedure using a variety of publicly available datasets and models. We also validate our procedure using techniques from interpretable learning and feature selection.

1 Introduction

Machine-learning models now determine and control an increasing number of real-world decisions. Judges use recidivism prediction tools to guide sentencing, and would want to know that race did not play a role in a sentencing guideline produced by a machine-learning model. Chemists using computer methods to predict the outcome of chemical experiments may want to know the extent to which a specific chemical property determines the outcome. But as these models have become more powerful and ubiquitous, they have become more opaque. Deep learning systems are a notable example of highly successful and yet highly complex models that are sensitive to train and have millions of parameters that are hard to understand. This presents a challenge. How can we *audit* such models to gain a deeper understanding? As model creators, it can be useful 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

*This research was funded in part by the NSF under grants IIS-1251049, CNS-1302688, IIS-1513651, and DMR-1307801.

influenced by certain features. For example, in answer to the judge's question about whether race played a role in sentencing guidelines, the judge may be interested to know whether it was more or less important to the guidelines than the number of prior arrests, and may also be interested in a quantification of the amount race contributed to the outcomes. A public defender (as an outside auditor) might also be interested in the pattern of decisions by the system.

Building interpretable models is one solution. But even if we could create algorithms that are simple and models that are interpretable, the proprietary nature of many of these predictive systems, often with algorithms accessible only by API, means that we need to develop infrastructure to examine and audit *black-box models*: systems to which an auditor has only limited access.

1.1 Our Work

In this paper, we develop a technique to do black-box auditing of classification models. The output of our audit will be a ranking of features based on their influence on the outcome (this goal will be developed more precisely in Section 2).

What does it mean for a feature to influence the outcome of a model? We might take the approach, as feature selection does, that the presence or absence of the feature alone, and its impact on the model's accuracy, should determine its importance. Yet this may not adequately address the judge's concern: if zip code is being used as a proxy variable in order to make discriminatory decisions, a correct auditing would indicate that race (indirectly, through zip code) helped to determine the model's outcome. Rather than trying to model influence explicitly via information-theoretic or statistical correlations, we take a functional approach: *the information content of a feature can be estimated by trying to* predict *it from the remaining features.*

Suppose we cannot predict a feature from the rest of the data. Then clearly it contains information that is unique. But is it relevant to the task at hand? This brings up the second part of our approach. Once we have eliminated the influence of a feature on the rest of the data by ensuring that it cannot be predicted, we can then test the model with the newly *obscured* data set. Any decrease in accuracy can then be attributed to the feature we eliminated.

Contributions Our main contributions include

- A technique to *obscure* the influence of a feature on an outcome.
- A method for rank-ordering features 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.

1.2 Alternative Approaches

While fields outside of computer science have begun paying attention to the idea of auditing black-box machine learning algorithms [15, 5], within computer science this question has mostly been addressed separately from machine learning in the domain of auditing black-box web applications for security vulnerabilities [2] or in validating known (non black-box) models [11]. Our work is related to, but is distinct from, a number of different approaches that address larger issues of interpretability and model understanding. We briefly discuss these approaches here and present a detailed review of the literature in Section 5.

Feature selection – the process of identifying features that have significant influence on the outcome – is the most related technique to black-box auditing. A crucial difference is that wrapper

methods for feature selection work by identifying the degree to which a *retrained* model's behavior changes if the feature is changed or removed. In black-box auditing we do not have the ability to retrain the model. Additionally, eliminating a feature might not remove its influence because of correlated attributes: this is something we address directly. Nevertheless, we use feature selection as a baseline comparison to evaluate our methods.

Instead of auditing black-box models, one could build models that are *interpretable*. Models created with the goal of interpretability assume direct (i.e non-black-box) access to the model and do not attempt to create a feature ranking. In fact, we will show in Section 4 that interpretable models can be used as a separate audit to confirm our black-box auditing technique.

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 [6]. 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.

2 Black-box Feature Auditing

We begin by formally introducing the *black-box feature auditing problem*.

Let $f\colon \mathbb{X} \to \hat{Y}$ be a black-box binary classification function, where $\mathbb{X}$ is a set of feature vectors and $\hat{Y}$ is an outcome vector. Let Y denote the correct outcomes, $f(\mathbb{X}) = \hat{Y}$ denote the predicted outcomes, and denote the accuracy of prediction as $\texttt{acc}(Y, \hat{Y})$. We will also be interested in the capacity to predict a specific feature vector $X_i \in \mathbb{X}$ from a modified version of $\mathbb{X}$ that has been *obscured* with respect to feature X_i. We will describe the obscuring process in the following section, but will for now denote the set of features with X_i obscured as $\hat{\mathbb{X}} \setminus X_i$. We will say that feature X_i has been *removed* from $\hat{\mathbb{X}} \setminus X_i$. In order to define "obscuring" formally, we will examine the class-conditioned version of the error rate when attempting to predict X_i:

Definition 2.1 (BER)**.** *Let $f : \hat{\mathbb{X}} \setminus X_i \to X_i$ be a predictor of X_i from $\hat{\mathbb{X}} \setminus X_i$. The* balanced error rate BER *of f on distribution $\mathcal{D}$ over the pair $(\hat{\mathbb{X}} \setminus X_i, X_i)$ is defined as the (unweighted) average class-conditioned error of f. In other words,*

$$\text{BER}(f(\hat{\mathbb{X}} \setminus X_i), X_i) =$$

$$\frac{\Pr[f(\hat{\mathbb{X}} \setminus X_i) = 0 | X_i = 1] + \Pr[f(\hat{\mathbb{X}} \setminus X_i) = 1 | X_i = 0]}{2}$$

Definition 2.2 (Predictability)**.** *X_i is said to be* ϵ-predictable *from $\hat{\mathbb{X}} \setminus X_i$ if there exists a function $f : \hat{\mathbb{X}} \setminus X_i \to X_i$ such that*

$$\text{BER}(f(\hat{\mathbb{X}} \setminus X_i), X) \leq \epsilon.$$

Finally, we can define what we mean to obscure a feature.

Definition 2.3 (ϵ-obscure)**.** *We define $\hat{\mathbb{X}} \setminus_\epsilon X_i$ as the* ϵ-obscure *version of $\mathbb{X}$ with respect to a specific feature X_i if X_i is* not *ϵ-predictable from $\hat{\mathbb{X}} \setminus_\epsilon X_i$. I.e., if, for all functions $f\colon \hat{\mathbb{X}} \setminus X_i \to X_i$,*

$$\text{BER}(f(\hat{\mathbb{X}} \setminus X_i), X_i) > \epsilon \ .$$

We can now give a formal definition of the black-box feature auditing problem.

Problem 2.1 (Black-box Feature Auditing)**.** *Given a black-box binary classification function $f\colon \mathbb{X} \to \hat{Y}$ with correct outcomes Y, output a rank ordering of feature vectors X_1, X_2, ..., X_n such that for all $i < j$, $\texttt{acc}(f(\hat{\mathbb{X}} \setminus_\epsilon X_i), Y) < \texttt{acc}(f(\hat{\mathbb{X}} \setminus_\epsilon X_j), Y)$ where $\hat{\mathbb{X}} \setminus_\epsilon X_i$ is the ϵ-obscured version of $\mathbb{X}$ with respect to X_i (and X_j respectively).*

Notes The definition of predictability is borrowed from [6]. The definition of ϵ-obscurity is same as the definition of ϵ-fairness from that work, but applied to any feature, rather than just "protected" ones.

3 Gradient feature auditing

In this section, we will introduce a method we call *gradient feature auditing* to solve the black-box feature auditing problem. Informally, gradient feature auditing gradually removes each individual feature from a dataset in order to assess how useful that feature was for the original model. More precisely, for each feature X_i, we construct the ϵ-obscured data set $\hat{\mathbb{X}} \setminus X_i$ and compare the accuracy $\texttt{acc}(f(\mathbb{X}), Y)$ with the accuracy obtained by classifying the *obscured* data set $\hat{\mathbb{X}} \setminus X_i$. We then rank-order all features by the difference between these two accuracies. The main algorithm is summarized in Algorithm 1[1].

Algorithm 1 Gradient feature auditing

Input: Data $\mathbb{X}$, labels Y, obscurity parameter ϵ.

procedure GradientFeatureAudit(f, $\mathbb{X}$, Y, ϵ)
  base_acc $\leftarrow$ acc$(f(\mathbb{X}), Y)$
  for each $X_i \in \mathbb{X}$ **do**
    $\hat{\mathbb{X}} \setminus_\epsilon X_i =$ Obscured$(\mathbb{X}, X_i, \epsilon)$
    $\hat{Y} \leftarrow f(\hat{\mathbb{X}} \setminus_\epsilon X_i)$
    map$[name(X_i)] \leftarrow$ base_acc $-$ acc$(\hat{Y}, Y)$
  end for
  feature_list $\leftarrow$ keys(**sort** map decreasing by values)
  return feature_list
end procedure

Varying this gradient feature auditing algorithm over different values of ϵ for the ϵ-obscured versions of $\mathbb{X}$, we expect to see a degradation in the predictive ability of the model f.

3.1 ϵ-obscurity

In order to calculate the ϵ-obscured version of $\mathbb{X}$ with respect to X_i recall from Definition 2.3 that we must show that X_i is not ϵ-predictable from $\hat{\mathbb{X}} \setminus X_i$ by any function $f\colon \hat{\mathbb{X}} \setminus X_i \to X_i$.

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

Define the median distribution A such that its cumulative distribution F_A is given by

$$F_A^{-1}(u) = \text{median}_{w \in W} F_x^{-1}(u)$$

In [6] 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 X is maximally obscured, but W also minimally changed, in that A also minimizes the function $\sum_{w \in W} d(W_x, A)$ where $d(\cdot, \cdot)$ was the earthmover distance between the distributions using ℓ_2 as the base metric. It was further shown that we could modify W *partially* by moving values of W only partially towards the destination. This partial

[1] Full code can be found at `https://github.com/cfalk/BlackBoxAuditing`

movement is parametrized by a parameter $0 \leq \lambda \leq 1$ that can be viewed as a monotonic function of the obscurity parameter ϵ (where $\lambda = 0$ represents the original values and $\lambda = 1$ represents moving W so it completely mimics A).

However, if attributes to be obscured and removed are not numerical and categorical respectively, the above procedure does not work. In what follows, we describe how we address both issues. Our algorithm is summarized in Algorithm 2.

Algorithm 2 Obscuring feature set $\mathbb{X}$ with respect to $X_i \in \mathbb{X}$

procedure OBSCURED($\mathbb{X}$, X_i, λ)
  $\hat{\mathbb{X}} \setminus X_i \leftarrow \emptyset$
  if X_i is numerical **then**
    $X_i \leftarrow$ `Binned`(X_i)
  end if
  for each $X_j \in \mathbb{X} \setminus X_i$ **do**
    if X_j is numerical **then**
      $\hat{X}_j \leftarrow$ `ObscureNumerical`(X_j, X_i, λ)
    else
      $\hat{X}_j \leftarrow$ `ObscureCategorical`(X_j, X_i, λ)
    end if
    $\hat{\mathbb{X}} \setminus X_i \leftarrow \hat{\mathbb{X}} \setminus X_i \cup \hat{X}_j$
  end for
  return $\hat{\mathbb{X}} \setminus X_i$
end procedure

3.2 Removing numerical attributes

In order to remove a numerical attribute, 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, it is reasonable to bin the numerical attribute and use the bins as categories in the original obscuring procedure. Bins are chosen according to the Freedman-Diaconis rule, used for choosing histogram bin sizes [7].

3.3 Obscuring categorical attributes

In the categorical setting, 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 $\sum_{x \in X} d(W_x, A)$. 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

$$d(p, q) = \|p - q\|_1 \ .$$

It thus follows that 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 $p_A(w) = \text{median}_x W_x(w)$. Once such an A has been computed, we can find the exact repair by computing the earthmover distance (via min-cost flows) between each W_x and A. By using min-cost flows to do the repair, and not just changing values arbitrarily in order to achieve A, we aim to ensure that the changes are minimal.

Since we must have an integral number of observations in each category in the resulting obscured attribute, there are a few remaining details to handle. We must create the obscured version of W, denoted $\hat{W}$, so as to ensure that $|\{\hat{W}|X = x\}| \in \mathbb{Z}$ for all values of $x \in X$. Let $|\hat{W}_{w,x}|$ denote $|\{\hat{W} = w|X = x\}|$. We set

$$|\hat{W}_{w,x}| = \lfloor p_A(w) \cdot |\{W|X = x\}| \rfloor \ .$$

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 circulation problem solution describes how to distribute observations at a per category, per obscured attribute value level. Individual observations within these per category, per attribute buckets can be distributed arbitrarily. The observations that flow to the lost observations node are distributed randomly according to distribution A.

4 Experiments

In order to evaluate the introduced gradient feature auditing algorithm, we will consider experiments on the following five data sets, chosen to balance easy replicability with demonstration on domains where these techniques are of practical interest.

Synthetic data. We created synthetic data with 6,000 items where the first half of the data was assigned to the first class and the second half to the second. Features included directly encode the row number (features A, B, and C), are constant, or are random. We use a random two-thirds / one-third training / test split on this data.

Adult Income and German Credit data. We will consider two data sets from the UC Irvine machine learning repository[2] that are of frequent use in the algorithmic fairness literature, comparing fairness (obscurity, in our context) to accuracy. The first of those is the Adult Income data set which consists 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. 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 two-thirds / one-third training / test split on this data and the two data sets described below.

Recidivism data. The Recidivism Prediction data set is taken from the National Archive of Criminal Justice Data[3] and contains information on 38,624 prisoners sampled from those released from 15 states in 1994 and tracked for three years. The full data includes attributes describing the entirety of the prisoners' criminal histories before and after release as well as demographic data. For this work, we processed the data additionally to match the description of the processing performed in a previous study on interpretable models [18]. This processing resulted in 10 categorical attributes. The prediction problem considered is that of determining whether a released prisoner will be rearrested within three years. The version of the dataset used in this paper is available online.[4]

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

[3] `http://doi.org/10.3886/ICPSR03355.v8`

[4] `https://github.com/cfalk/BlackBoxAuditing`

Dark Reactions data. The Dark Reactions data set[5] is taken from historical hydrothermal synthetic procedures, recorded irrespective of the outcome of the procedure. The intended synthetic products are generally templated inorganic-organic hybrid materials, with a wide variety Amine compounds forming the organic component of the generated materials. Input variables are calculated as aggregate properties of ionic properties (in the case of the inorganic components) and semi-empirical properties in the case of the organic component. The response variable is a boolean value which is defined as the presence or absence of an ionic crystalline system in the product of the reaction, and is set to 1 or 0 respectively. There are 273 features for this dataset, which are calculated as the aggregates of various chemical properties of components of the chemical systems concerned, as well as the conditions under which the procedures were carried out. 5,496 such historical experiments are included in the data set.

Models. Since our goal is to examine the gradient feature auditing technique in the face of black-box models, we built two models that are notoriously opaque to simple examination; SVMs[6] [9] and Feedforward neural networks (FNNs).[7] We also include C4.5 decision trees[8] [13] so that the audited results can be examined directly in comparison to 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. The Recidivism data set FNN was trained using `softmax` activations on a fully connected hidden layer of 100 nodes. The model was trained using a batch size of 500 and 100 epochs.

4.1 Auditing using test data

The work of [6] assumes that $\hat{\mathbb{X}} \setminus X_i$ is used to *train* a function $f : \hat{\mathbb{X}} \setminus X_i \to \hat{Y}$ so that $\hat{Y}$ can be guaranteed to be ϵ-obscured with respect to X_i. However, in a black-box auditing context, we can not retrain the model with the obscured data set. So is $\hat{Y}$ obscured with respect to X_i if $f : \hat{\mathbb{X}} \setminus X_i \to \hat{Y}$ where $\hat{\mathbb{X}} \setminus X_i$ is the *test* data and the model f was trained on $\mathbb{X}$? We answer this question affirmatively, with empirical justification.

The intuition behind the obscuring process is that once an attribute W is obscured with respect to an attribute X, there is no way to learn about X given values of W. In [6], this property was used to argue that a model that is *trained* on the obscured data cannot pick up any information about X 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 X cannot take advantage of this information, because the presence of X is masked in the data through the obscuring process. Thus, the results of classification will be free of the influence of X.

We can empirically validate this claim by running an experiment where we *do* retrain the model

[5] `http://darkreactions.haverford.edu`

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

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

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

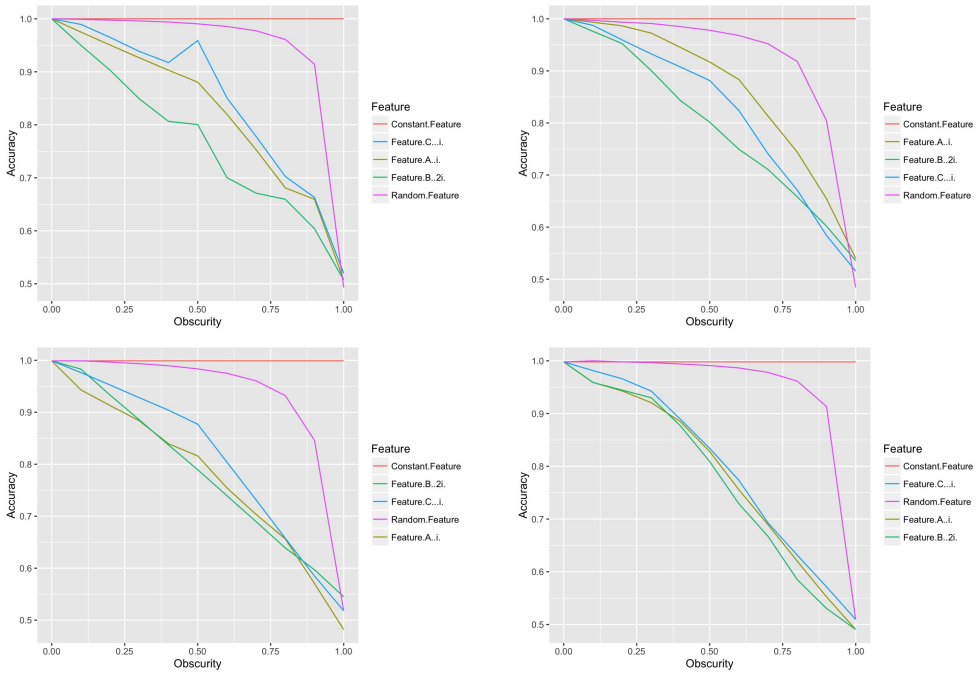

Fig. 1: Synthetic data that is retrained each time the obscuring procedure is run (first row) as compared to the same data with the gradient feature auditing method that obscures only the test data and does not retrain the model. The first column shows this with a decision tree and the second column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

(a la [6]) 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 repairing the test data (see Figure 1), thus validating the ability to repair directly on the test data without retraining the model.

4.2 Black-box feature auditing

In order to begin to assess the extent to which the gradient feature auditing method is auditing the model (and not just the features themselves), we trained each of the models (SVMs, FNNs, and C4.5 decision trees) on each of the five data sets. We then ran gradient feature auditing using the test data for each data set. Figure 2 shows the resulting gradient feature auditing plots, demonstrating the gradient for each feature as the amount that the data is obscured in terms of that feature increases and the accuracy decreases.

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 (which is based directly on the row index i determining each feature) causes the model to degrade to 50% accuracy as expected. 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 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. Thus, on the synthetic data, the resulting curve and associated feature ranking reflects our expectations.

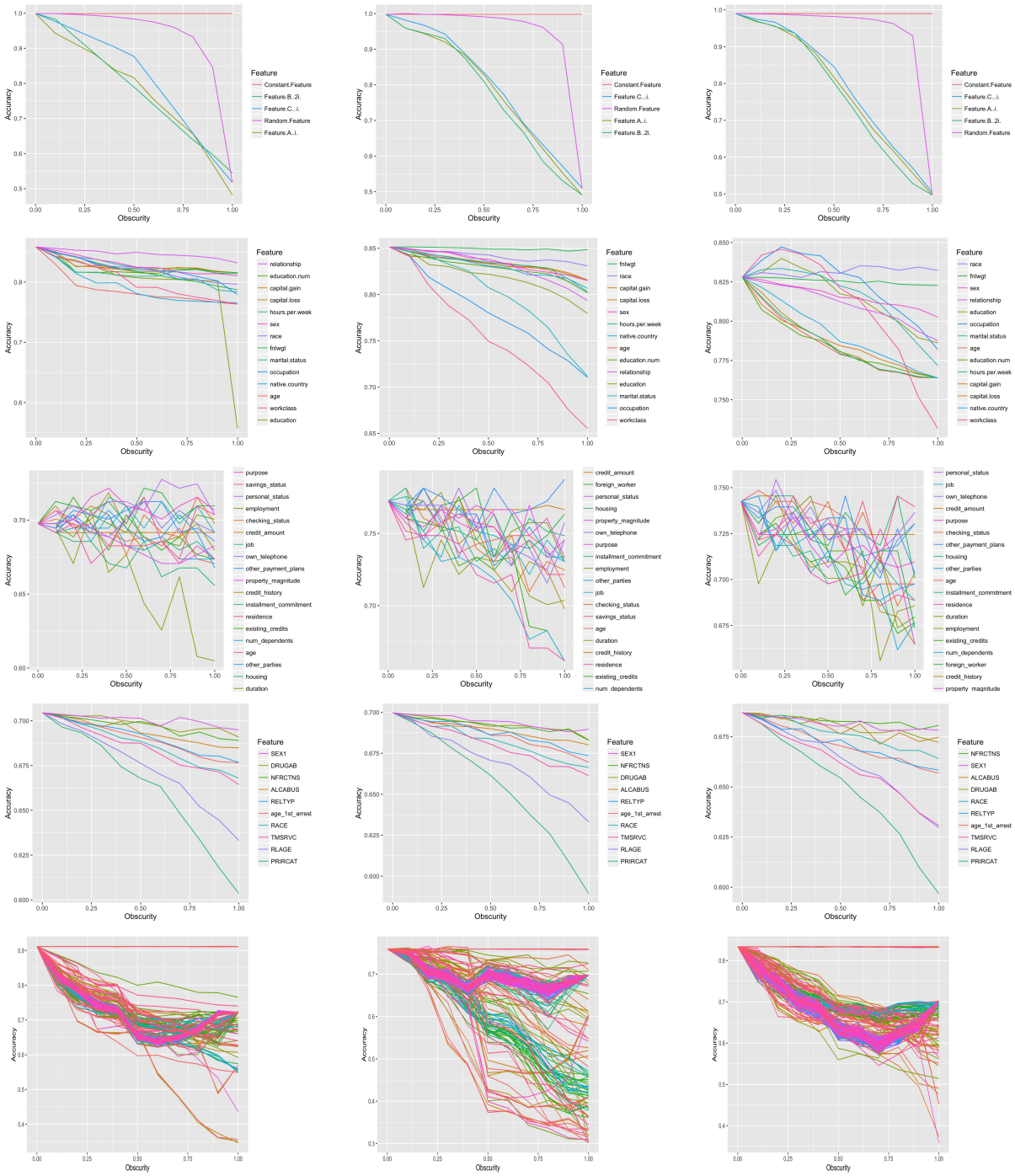

Fig. 2: Obscurity vs. accuracy plots for each model and each data set considered. First column: C4.5 decision trees. Second column: SVMs. Third column: FNNs. First row: Synthetic data. Second row: Adult income data set. Third row: German credit data. Fourth row: Recidivism data. Final row: Dark Reaction data, shown without a feature legend due to the large number of features.

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 towards the bottom of the charts. While `workclass` is found to be highly ranked by all models, the removal of `race` does not have much impact on the SVM or FNN models, but is used by the decision tree model. On the FNN model gradient auditing plot we also see that there was a set of features for which partially removing the feature information 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 seem to exhibit some arbitrary ordering. We hypothesize that models which are poor are likely to produce such responses when run through the gradient feature auditing technique. This is for two main reasons; firstly, the gradient feature auditing technique assesses the importance of a feature. If a model is already poor at making predictions, then attributes selected using this measure are going to be limited by that fact. Furthermore, a model which is inaccurate may also be predicated on an incorrect set of features or based on over-modeling of information within what is effectively random noise. This would give rise to apparently spurious and arbitrary responses to gradient feature auditing, as seen on the German credit data for all three models. In these contexts, it makes more sense to consider the change in accuracy under a consistency measure. We'll explore this further in the next section.

Recidivism data. On the Recidivism data we see incredible consistency between the rankings of the different models. The top three most important features under all three models are PRIRCAT, a categorical representation of the number of prior arrests of the prisoner, RLAGE, the age at release, and TMSRVC, the time served before the release in 1994. The four least important features are also common to all three models: the sex of the prisoner, whether the prisoner is an alcohol or drug abuser, and the number of infractions the prisoner was disciplined for while incarcerated.

Dark Reactions data. The Dark Reactions data shows different ranked importances for the decision tree and SVM models. In the decision tree case, the most important features are shown to be the presence/absence of Group 13 elements, the presence/absence of Gallium specifically (a group 13 element), the present of trivalently bonded Vanadium, the presence/absence of Phosphorous, and the Van der Waals surface area of the organic components in the reaction mixture. That the first four of these features describe the presence or absence of elements indicates a separation of the kinds of chemistry being examined. The organic Van der Waals radius may be similar in this regard, coding for the nature of the organic component in terms of its physical size. The latter would be considered a chemically important descriptor when examining chemical synthetic routes in a more traditional manner.

In the case of the SVM, two descriptors for chemical presence also appear in the top 5 descriptors, namely the presence/absence of Bromine, and the present or absence of tetravalent Phosphorous. These likely rank highly for the same reason as similar descriptors in the decision tree ranking. The 2nd and 3rd ranked features are Number of Inorganic Components and overall Number of Components. These may be acting as a proxy variable for the kinds of chemistry being performed, or serving as indicators for success since reactions which contain larger numbers of components are necessarily more complex and as such are thought to have lower success rates. That pH is the most 'important' descriptor using this metric fits with chemical theory, insofar as many properties of compounds are altered by their pH, not least of which is the kind of reactions that they undergo; further, a number of pH dependent descriptors are contained within this data are known to be correlated with the pH (they are chemically dependent upon it), and thus the removal of correlated components as performed in GFA would remove a larger proportion of information from the dataset than other descriptors, thus ranking it as high importance to the model.

In both cases, the lowest ranked descriptors pertain to descriptors which only have one class present in the data, and as such, we do not expect these descriptors to have any input into any model formed, which is in line with what is seen in this importance ranking.

4.3 Auditing for consistency

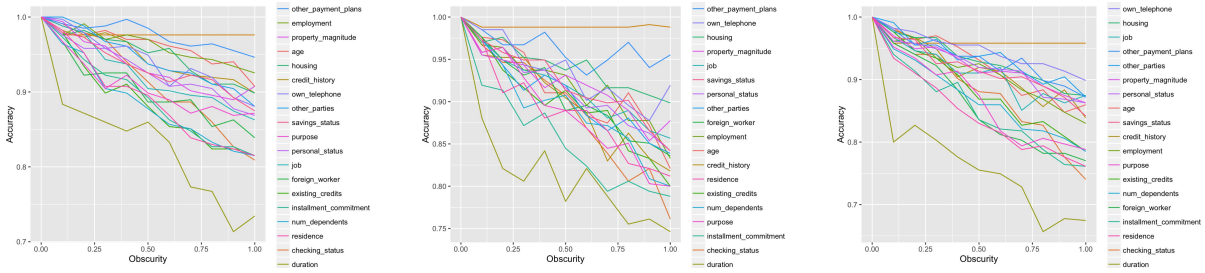

Fig. 3: Obscurity vs. consistency plots for the German Credit data. First column: decision tree model. Second column: SVM. Third column: FNN.

The results in Figure 2 serve a dual purpose - creating a feature ranking and judging from the results if a model under these conditions is still accurate. For most of the data sets, these two purposes can be satisfied at the same time, because the initial model is accurate enough that it degrades smoothly. As discussed earlier, however, the German Credit data set's initial model wasn't very accurate, and so there is a lot of effective noise in the results. We can remove this by measuring our gradient according to the accuracy with respect to the results predicted by the original model. We'll call this measure the *consistency* of a model. It effectively relabels the data so that the predictions of the original model are the new labels and calculates the accuracy as usual with respect to those labels. The unobscured data will thus always have a consistency of 100%. The new gradient measured will be the difference between the 100% consistency at obscurity of 0 and the degraded consistency when fully obscured.

As can be seen in Figure 3, under the consistency measure the accuracy of the model now degrades smoothly so that a ranking can be extracted. The slight noise still 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 `duration` the most important and `checking status` the second most important features.

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 Figure 2 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 Figure 4) we see that while the ranking derived from accuracy closely resembles the consistency ranking, a cluster of features (`native.country`, `capital.loss`, `capital.gain`, `hours.per.week`, `education.num`, and `age`) had been ranked above `occupation` and `marital.status` and the consistency ranking moves that cluster down in rank.

4.4 Comparison to feature selection

While gradient feature auditing (GFA) is fundamentally different from feature selection sincea) 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 gener-

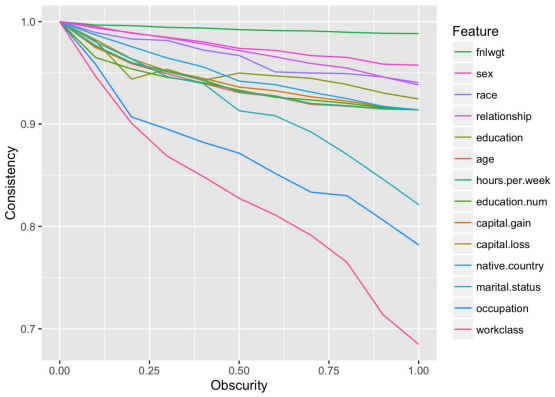

Fig. 4: Obscurity vs. consistency chart for the Adult Income data modeled by an FNN.

ated 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.[9] For both the Adult Income and German Credit data sets feature selection created identical rankings regardless of whether the 'wrapper' method was specified to rank for a decision tree or an SVM.

Spearman's rank correlation is a nonparametric test of the relationship between two variables. Specifically, it applies to situations in which the values for the variable are ranks, rather than continuous numerical values. In this case, the rank given is the ranked ordering by each method of feature selection. Each feature is granted a rank value, and the monotonic correlation (represented by the Spearman's correlation coefficient) of these values therefore describes the degree of similarity between two methods of feature selection.

The synthetic data had a strong correlation between the feature selection rankings and the GFA rankings, with feature A being ranked first by all methods. However, the number of features is so small in this case that it is not possible to ascribe statistical significance to this result. Correlation between the feature selection derived rankings from the Adult Income and German Credit data sets and the ranking of features as indicated by GFA was weak for both models. The correlations between feature selection ranks and GFA ranks for the German Credit data were an order of magnitude weaker than those for the other examined data sets. This stands to reason, since the models for this data were so poor, meaning that any ordering of the features is somewhat arbitrary. When the consistency ranking is used instead, the correlation with feature selection increases.

That there is a difference in this ordering is not unexpected, since the two methods ask philosophically different questions; in GFA, the assessment is being made as to how important a descriptor directly is for a specific instance of a model to make accurate predictions. In the case of feature selection, the question being asked is whether a descriptor will be useful for an as-yet uninstantiated model to accurately predict real-world outcomes. In addition, feature selection looks at each feature separately, and only examines features as they are directly included in models. GFA, examines not only the individual descriptors used to build the model instance, but also potential external proxy descriptors, and removes information from the overall dataset, not simply the influence of individual descriptors separately.

[9] 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

4.5 Modeling the model

Another point of comparison that we can use to determine if our gradient feature auditing technique is successfully representing the model's choices is to create an interpretable model of the model that we can examine by hand and create a comparison feature ranking from. By a "model of a model" we mean that we should 1) train the model M on training data $(\mathbb{X}, Y)$, 2) determine new labels $\hat{Y}$ from the predicted outcomes $M(\mathbb{X})$, and 3) overfit an interpretable model $I(M)$ to these predicted labels (see Algorithm 3 and [1]). (This idea is similar to model compression, but without the need to find new test data [3].) Assuming that the model resulting from this procedure has high accuracy on $\hat{Y}$, we now have an interpretable model of our model. We can now compare the features the interpretable models use to the results of our auditing procedure.

Algorithm 3 Modeling a model M, already trained on data $\mathbb{X}$, to create interpretable model $I(M)$.

1: **procedure** ModelAModel(M, $\mathbb{X}$)
2: $\quad\hat{Y} \leftarrow M(\mathbb{X})$
3: $\quad I(M) \leftarrow$ **train**$(I, \mathbb{X}, \hat{Y})$
4: $\quad$**return** $I(M)$
5: **end procedure**

For the SVM and decision tree models trained on the each of the five data sets, we trained a decision tree model of the model. For the decision tree original models, we use this to confirm the model of a 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 a ranking of features by the probability that they appeared on the path from the root node 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 appear first in the list.

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 results 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 over-fit 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). This reveals an important flaw with this simple probability ranking - it does not handle proxy variables, as required by the black-box feature auditing problem. Thus, its confirmation of the gradient feature auditing technique - which does appropriately rank proxy variables - will not be perfect. Indeed, on the synthetic data we see that features A, B, and C were all ranked highly by gradient feature auditing.

Evaluating the GFA rankings. To understand the extent to which the feature ranking described is similar to the one generated by our proposed obscuring algorithm, we compare the feature rankings of three datasets (Adult, Recidivism and German) to the gradient feature auditing ranking presented in Algorithm 1. This experiment is similar to the one that compares the GFA ranking to feature selection techniques as described in Section 4.4, but now we compare the rank ordering from GFA to the one produced by analyzing the model of models. Specifically, we collect the three generated rankings (one per dataset) into one vector for each of the feature ranking procedures, and run a single Spearman rank-correlation statistical test. We find a combined sample rank correlation

of 0.413 (a 95% bootstrap confidence interval of $[0.115, 0.658]$), and find also that we can reject the null hypothesis (of no correlation), with $p < 0.002$. This provides evidence that our feature auditing procedure correctly captures a sensible feature importance ordering. We note, in addition, that when we compare the GFA ranking based on model *consistency* (cf. Section 4.3), the results are similar to the ones based on model accuracy.

One might wonder now, if the rank ordering of the model-of-model *also* produces a feature ordering, why is gradient feature auditing necessary? The answer is simple, but bears repeating. The probability ranking of the model-of-model variables has several shortcomings. Critically, it cannot handle proxy variables and correlations, both key requirements of the black-box feature auditing problem. In other words, the model of models offers independent *validation* of GFA, but is not a replacement for it.

5 Related Work

Feature Selection. Feature selection is fundamentally different from the black-box feature auditing problem in that one of the main restrictions on the black-box feature auditing problem is that it may not change the set of features given as input to the algorithm (because it may be accessed via an API, for instance). Yet, given that black-box feature auditing creates a ranking of features based on importance to the model, the problems are also closely related. Feature selection methods can be broadly categorized as filter methods and wrapper methods (see survey [4]). Filter methods generate a feature ranking by evaluating the quality of a given feature subset with respect to a measure independent of the classifier (e.g., the mutual information of the features), while wrapper methods evaluate the ranking with respect to the quality of the prediction. A ubiquitous wrapper 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, generally measured using some correlation coefficient between predicted and actual outcomes and the frequentist certainty thereof - a measure which can apply equally well to black box methods. In practice and principle, this is highly analogous to feature auditing with the exception of the use of certainty statistics, and the constraint placed upon feature auditing already stated.

Interpretability. Model interpretability is an area that aims to create models that are sparse, meaningful, and intuitive and thus human-understandable by experts in the field the data is derived from [16]. The classic example of such models are decision trees [13], while recently supersparse linear integer models (SLIMs) have been developed as an alternative interpretable model for classification [16].

Model understanding. Deep learning models are notoriously not interpretable. Recent attempts at understanding how these make decisions on image inputs have included visualizing the derived representation by reversing a neural net to output an image when given a label [17], probing nodes to determine what stimulus image produces the highest activation [12], and examining activation patterns across local input neighborhoods [10]. Typically, the goal in this subfield is to understand the models in order to improve their accuracy. While these results are not directly applicable to our problem, the issue of model opacity serves as motivation for this work.

Algorithmic fairness. The area of algorithmic fairness, which develops techniques to prevent machine learning algorithms from making discriminatory decisions, can be roughly categorized into algorithms that preprocesses the input, modifications to machine learning algorithms themselves so that they make fair decisions, and algorithms that post process the output so that it is fair (see survey [14]). While these techniques may initially seem unrelated to the feature auditing question, the algorithms in the first category aim to prevent any machine learning algorithm from using a specific feature of the training data (e.g., race) or any variables correlated with it (i.e., proxy

variables such as zip code) to determine the outcome. As we showed here, such techniques can be used to obscure data.

References

[1] N. Barakat and J. Diederich. Learning-based rule-extraction from support vector machines. In *Proc. of the 14th International Conference on Computer Theory and Applications*, 2004.

[2] Jason Bau, Elie Bursztein, Divij Gupta, and John Mitchell. State of the art: Automated black-box web application vulnerability testing. In *Proc. of the IEEE Symp. on Security and Privacy*, pages 332–345, 2010.

[3] Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541. ACM, 2006.

[4] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers and Electrical Engineering*, 40:16–28, 2014.

[5] Nicholas Diakopoulos. Algorithmic accountability reporting: on the investigation of black boxes. *Tow Center for Digital Journalism*, Feb. 2014.

[6] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

[7] David Freedman and Persi Diaconis. On the histogram as a density estimator: L 2 theory. *Probability theory and related fields*, 57(4):453–476, 1981.

[8] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[9] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. MIT Press, 1998.

[10] Mayank Kabra, Alice Robie, and Kristin Branson. Understanding classifier errors by examining influential neighbors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3917–3925, 2015.

[11] Jack P. C. Kleijnen. Verification and validation of simulation models. *European Journal of Operational Research*, 82:145–162, 1995.

[12] Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng. Building high-level features using large scale unsupervised learning. In *Proc. of the International Conference on Machine Learning*, 2011.

[13] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[14] Andrea Romei and Salvatore Ruggieri. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, pages 1–57, April 3 2013.

[15] Christian Sandvig, Kevin Hamilton, Karrie Karahalios, and Cedric Langbort. An algorithm audit. In Seeta Pena Gangadharan with Virginia Eubanks and Solon Barocas, editors, *Data and Discrimination: Collected Essays*. Open Technology Institute, New America, 2014.

[16] Berk Ustun, Stefano Traca, and Cynthia Rudin. Supersparse linear integer models for interpretable classification. Technical Report 1306.6677, arXiv, 2014.

[17] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision — ECCV 2014*, pages 818–833. Springer, 2014.

[18] Jiaming Zeng, Berk Ustun, and Cynthia Rudin. Interpretable classification models for recidivism prediction. Technical Report 1503.07810, arXiv, 2015.