

# How can we fool LIME and SHAP?

## Adversarial Attacks on Post hoc Explanation Methods

Dylan Slack<sup>1,\*</sup> Sophie Hilgard<sup>2,\*</sup> Emily Jia<sup>2</sup> Sameer Singh<sup>1</sup> Himabindu Lakkaraju<sup>2</sup>

<sup>1</sup>University of California, Irvine

<sup>2</sup>Harvard University

### Abstract

As machine learning black boxes are increasingly being deployed in domains such as healthcare and criminal justice, there is growing emphasis on building tools and techniques for explaining these black boxes in an interpretable manner. Such explanations are being leveraged by domain experts to diagnose systematic errors and underlying biases of black boxes. In this paper, we demonstrate that post hoc explanations techniques that rely on input perturbations, such as LIME and SHAP, are not reliable. Specifically, we propose a novel *scaffolding* technique that effectively hides the biases of any given classifier by allowing an adversarial entity to craft an arbitrary desired explanation. Our approach can be used to scaffold any biased classifier in such a way that its predictions on the input data distribution still remain biased, but the post hoc explanations of the scaffolded classifier look innocuous. Using extensive evaluation with multiple real world datasets (including COMPAS), we demonstrate how extremely biased (racist) classifiers crafted by our framework can easily fool popular explanation techniques such as LIME and SHAP into generating innocuous explanations which do not reflect the underlying biases.

### Introduction

Owing to the success of machine learning (ML) models, there has been an increasing interest in leveraging these models to aid decision makers (e.g., doctors, judges) in critical domains such as healthcare and criminal justice. The successful adoption of these models in domain-specific applications relies heavily on how well decision makers are able to understand and trust their functionality (Doshi-Velez and Kim 2017; Lipton 2016). Only if decision makers have a clear understanding of the model behavior, can they diagnose errors and potential biases in these models, and decide when and how much to rely on them. However, the proprietary nature and increasing complexity of machine learning models makes it challenging for domain experts to understand these complex *black boxes*, thus, motivating the need for tools that can explain them in a faithful and interpretable manner.

As a result, there has been a recent surge in post hoc techniques for explaining black box models in a human interpretable manner. One of the primary uses of such explanations is to help domain experts detect discriminatory biases in black box models (Tan et al. 2018; Kim et al. 2017). Most prominent of these techniques include *local, model-agnostic* methods that focus on explaining individual predictions of a given black box classifier, including LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017). These methods estimate the contribution of individual features towards a specific prediction by generating perturbations of a given instance in the data and observing the effect of these perturbations on the output of the black-box classifier. Due to their generality, these methods have been used to explain a number of classifiers, such as neural networks and complex ensemble models, and in various domains ranging from law, medicine, finance, and science (Elshawi, Al-Mallah, and Sakr 2019; Ibrahim et al. 2019; Whitmore, George, and Hudson 2016). However, there has been little analysis of the reliability and robustness of these explanation techniques, especially in the adversarial setting, making their utility for critical applications unclear.

In this work, we demonstrate significant vulnerabilities in post hoc explanation techniques which can be exploited by an adversary to generate classifiers whose post hoc explanations can be arbitrarily controlled. More specifically, we develop a novel framework that can effectively mask the discriminatory biases of any black box classifier. Our approach exploits the fact that post hoc explanation techniques such as LIME and SHAP are perturbation-based, to create a *scaffolding* around any given biased black box classifier in such a way that its predictions on input data distribution remain biased, but its behavior on the perturbed data points is controlled to make the post hoc explanations look completely innocuous. For instance, using our framework, we generate highly discriminatory classifiers (such as the ones that *only* use race to make their decisions) whose post hoc explanations (generated by LIME and SHAP) make them look completely innocuous, effectively hiding their discriminatory biases.

We evaluate the effectiveness of the proposed framework on multiple real world datasets - COMPAS (Larson et al. 2016), Communities and Crime (Redmond 2011), and Ger-

\*first two authors contributed equally to the paper.

Author Emails: dslack@uci.edu, ash798@g.harvard.edu, ejia@college.harvard.edu, sameer@uci.edu, hlakkaraju@seas.harvard.edu

man loan lending (Asuncion and Newman 2007). For each dataset, we craft classifiers that heavily discriminate based on protected attributes such as race (demographic parity ratio = 0), and show that our framework can effectively hide their biases. In particular, our results show that the explanations of these classifiers generated using off-the-shelf implementations of LIME and SHAP do not flag *any* of the relevant sensitive attributes (e.g., race) as important features of the classifier, thus demonstrating that the adversarial classifiers successfully fooled these explanation methods. These results suggest that it is possible for malicious actors to craft adversarial classifiers that are highly discriminatory, but can effectively fool existing post hoc explanation techniques. This further establishes that existing post hoc explanation techniques are not sufficiently robust for ascertaining discriminatory behavior of classifiers in sensitive applications.

## Building Adversarial Classifiers to Fool Explanation Techniques

In this section, we discuss our framework for constructing adversarial classifiers (*scaffoldings*) that can fool post hoc explanation techniques which rely on input perturbations. We first provide a detailed overview of popular post hoc explanation techniques, namely, LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017), and then present our framework for constructing adversarial classifiers.

### Background: LIME and SHAP

While simpler classes of models (e.g., linear models, decision trees) are often readily understood by humans, the same is not true for complex models (e.g., ensemble methods, deep neural networks). Such complex models are essentially black boxes for all practical purposes. One way to understand them is to build simpler *explanation models* that are interpretable approximations of these black boxes.

To this end, several techniques have been proposed in existing literature. LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017) are two popular *model-agnostic, local explanation* approaches designed to explain any given black box classifier. These methods explain individual predictions of any classifier in an interpretable and faithful manner, by learning an interpretable model (e.g., linear model) locally around each prediction. More specifically, LIME and SHAP estimate feature attributions for individual instances, which capture the effect/contribution of each feature on the black box prediction. Below, we provide some details of these approaches, while also highlighting how they relate to each other.

Let  $\mathcal{D}$  denote the input dataset of  $N$  data points i.e.,  $\mathcal{D} = (\mathcal{X}, \mathbf{y}) = \{(x_1, y_1), (x_2, y_2) \dots (x_N, y_N)\}$  where  $x_i$  is a vector that captures the feature values of data point  $i$ , and  $y_i$  is the corresponding class label. Let there be  $M$  features in the dataset  $\mathcal{D}$  and let  $\mathcal{C}$  denote the set of class labels in  $\mathcal{D}$  i.e.,  $y_i \in \mathcal{C}$ . Let  $f$  denote the black box classifier that takes a data point as input and returns a class label i.e.,  $f(x_i) \in \mathcal{C}$ . The goal here is to explain  $f$  in an interpretable and faithful manner. Note that neither LIME nor SHAP assume any knowledge about the internal workings of  $f$ . Let  $g$  denote

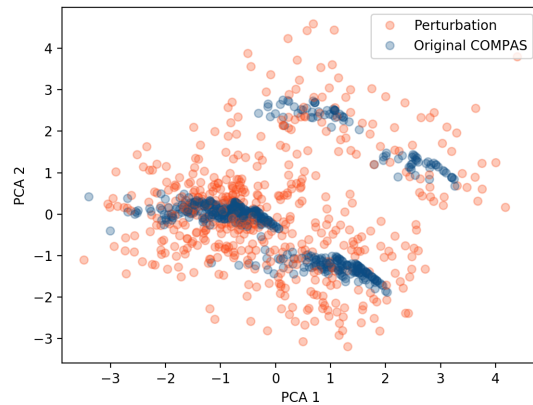


Figure 1: PCA applied to the COMPAS dataset (blue) as well as its LIME style perturbations (red). Even in this low-dimensional space, we can see that data points generated via perturbations are distributed very differently from instances in the COMPAS data. In this paper, we exploit this difference to craft adversarial classifiers.

an explanation model that we intend to learn to explain  $f$ .  $g \in \mathcal{G}$  where  $\mathcal{G}$  is the class of linear models.

Let the complexity of the explanation  $g$  be denoted as  $\Omega(g)$  (complexity of a linear model can be measured as the number of non-zero weights), and let  $\pi_x(x')$  denote the proximity measure between inputs  $x$  and  $x'$ , to define the vicinity (neighborhood) around  $x$ . With all this notation in place, the objective function for both LIME and SHAP is crafted to generate an explanation that: (1) approximates the behavior of the black box accurately within the vicinity of  $x$ , and (2) achieves lower complexity and is thereby interpretable.

$$\arg \min_{g \in \mathcal{G}} L(f, g, \pi_x) + \Omega(g) \quad (1)$$

where the loss function  $L$  is defined as:

$$L(f, g, \pi_x) = \sum_{x' \in X'} [f(x') - g(x')]^2 \pi_x(x')$$

The set  $X'$  corresponds to a set of inputs constituting the neighborhood of  $x$ .

The primary difference between LIME and SHAP lies in how  $\Omega$  and  $\pi_x$  are chosen. In LIME, these functions are defined heuristically:  $\Omega(g)$  is the number of non-zero weights in the linear model and  $\pi_x(x')$  is defined using cosine or  $l2$  distance. On the other hand, (Kernel) SHAP grounds these definitions in game theoretic principles to guarantee that the explanations satisfy certain desired properties. More details about the intuition behind the definitions of these functions and their computation can be found in Ribeiro, Singh, and Guestrin (2016) and Lundberg and Lee (2017).

## Our Proposed Framework

In this section, we discuss our framework in detail. First, we discuss some preliminary details about our set up. Then, we discuss the intuition behind our approach. Lastly, we present the technical details of our approach along with a discussion of some of our design choices and implementation details.

## Preliminaries

*Setting:* Assume that there is an adversary with an incentive to deploy a biased classifier  $f$  for making a critical decision (e.g., parole, bail, credit) in the real world. The adversary must provide black box access to customers and regulators (Regulation 2016), who may use post hoc explanation techniques to better understand  $f$  and determine if  $f$  is ready to be used in the real world. If customers and regulators detect that  $f$  is biased, they are not likely to approve it for deployment. The goal of the adversary is to fool post hoc explanation techniques and hide the underlying biases of  $f$ .

*Input:* The adversary provides the following inputs to our framework: 1) the biased classifier  $f$  which he/she intends to deploy in the real world and, 2) an input dataset  $\mathcal{X}$  which is sampled from the real world input data distribution  $\mathcal{X}_{dist}$  on which  $f$  will be applied. Note that neither our framework nor the adversary has access to  $\mathcal{X}_{dist}$ .

*Output:* The output of our framework will be a scaffolded classifier  $e$  (referred to as *adversarial classifier* henceforth) which behaves exactly like  $f$  when making predictions on instances sampled from  $\mathcal{X}_{dist}$ , but will not reveal the underlying biases of  $f$  when probed with leading post hoc explanation techniques such as LIME and SHAP.

**Intuition** As discussed in the previous section, LIME and SHAP (and several other post hoc explanation techniques) explain individual predictions of a given black box model by constructing local interpretable approximations (e.g., linear models). Each such local approximation is designed to capture the behavior of the black box within the neighborhood of a given data point. These neighborhoods constitute synthetic data points generated by perturbing features of individual instances in the input data. However, instances generated using such perturbations could potentially be off-manifold or out-of-distribution (OOD) (Mittelstadt, Russell, and Wachter 2019).

To better understand the nature of the synthetic data points generated via perturbations, we carried out the following experiment. First, we perturb input instances using the approach employed by LIME (See previous section). We then run principal component analysis (PCA) on the combined dataset containing original instances as well as the perturbed instances, and reduce the dimensionality to 2. As we can see from Figure 1, the synthetic data points generated from input perturbations are distributed significantly differently from the instances in the input data. This result indicates that detecting whether a data point is a result of a perturbation or not is not a challenging task, and thus approaches that rely heavily on these perturbations, such as LIME, can be *gamed*.

This intuition underlies our proposed approach. By being able to differentiate between data points coming from the input distribution and instances generated via perturbation, an adversary can create an adversarial classifier (*scaffolding*) that behaves like the original classifier (perhaps be extremely discriminatory) on the input data points, but behaves arbitrarily differently (looks unbiased and *fair*) on the perturbed instances, thus effectively fooling LIME or SHAP into generating innocuous explanations. Next, we formalize this

intuition and explain our framework for building adversarial classifiers that can fool explanation techniques.

**Building Adversarial Classifiers** Let  $f$  be the biased classifier described earlier. The adversary would like to design a framework such that if and when end users generate explanations of this black box, post hoc techniques can be fooled into thinking that the model is innocuous.

Recall that the real world data on which this classifier is likely to be applied follows a distribution  $\mathcal{X}_{dist}$ , and  $\mathcal{X}$  is a set of  $N$  data points sampled from this distribution that the adversary has access to. To fool the post hoc techniques, the adversary could design an adversarial classifier that exhibits biased behavior on instances sampled from  $\mathcal{X}_{dist}$ , and remain unbiased on instances that do not come from  $\mathcal{X}_{dist}$ . Since the feature importances output by LIME and SHAP rely heavily on perturbed instances (which may typically be OOD samples, e.g. Figure 1), the resulting explanations will make the classifier designed by the adversary look innocuous.

Assuming  $\psi$  is the classifier that is unbiased (e.g., makes predictions based on innocuous features that are uncorrelated with protected attributes), the adversarial classifier  $e$  takes the following form (See Algorithm 2 in Appendix):

$$e(x) = \begin{cases} f(x), & \text{if } x \in \mathcal{X}_{dist} \\ \psi(x), & \text{otherwise} \end{cases}$$

In order to create such a classifier, we need to be able to decide whether a given data point  $x$  comes from  $\mathcal{X}_{dist}$  or not; we discuss how to build such a classifier next.

**Detecting OOD Samples** To build a classifier `is_OOD` that detects if a given data point is an *out-of-distribution* (OOD) sample (`is_OOD(x) = True`) or not, we construct a new dataset from the instances in  $\mathcal{X}$ . Specifically, we *perturb* (more details in Experimental Evaluation) each instance  $x$  in  $\mathcal{X}$  to generate a new instance  $x_p$ . Let us denote the set of all the instances generated via perturbation as  $\mathcal{X}_p$ . The instances in  $\mathcal{X}$  are then assigned the class label `False` indicating that they are not OOD samples, while the instances in  $\mathcal{X}_p$  are assigned the class label `True` (indicating that they are OOD samples) unless they are very similar to the instances in  $\mathcal{X}$ . We then train an off-the-shelf classifier on the combined dataset  $\mathcal{X} \cup \mathcal{X}_p$  and their corresponding class labels (assigned as discussed above). Please refer to Algorithm 1 in Appendix for full pseudocode of `is_OOD`.

## Experimental Results

In this section, we discuss the detailed experimental evaluation of our framework. First, we analyze the effectiveness of the adversarial classifiers generated by our framework. More specifically, we test how well these classifiers can mask their biases by fooling multiple post hoc explanation techniques. Next, we evaluate the robustness of our adversarial classifiers by measuring how their effectiveness varies with changes to different parameters (e.g., weighting kernel, background distribution). Lastly, we present examples of post hoc explanations (both LIME and SHAP) of individual instances in the data to demonstrate how the biases of the classifier  $f$  are successfully hidden.

**Datasets** We experimented with multiple datasets pertaining to diverse yet critical real world applications such as recidivism risk prediction, violent crime prediction, and credit scoring. Below, we describe these datasets in detail (See Table 2 in Appendix for detailed statistics).

Our first dataset is the **COMPAS** dataset which was collected by ProPublica (Angwin et al. 2016). This dataset captures detailed information about the criminal history, jail and prison time, demographic attributes, and COMPAS risk scores for 6172 defendants from Broward County, Florida. The sensitive attribute in this dataset is race – 51.4% of the defendants are African-American. Each defendant in the data is labeled either as high-risk or low-risk for recidivism.

Our second dataset is **Communities and Crime (CC)** that captures various socio-economic and law enforcement aspects, as well as crime across various communities in the US (Redmond and Baveja 2002). This dataset contains information about 1994 communities (each community is a data point) in total. The sensitive attribute in this dataset is the percentage of Caucasian population. This percentage varies widely across communities and ranges from 2.68% to 99.63%. Each community (data point) is assigned a label depending on if the violent crime proportion in that community is above the median (violent crime rampant) or not.

The third dataset we use is the **German credit** dataset from the UCI repository (Blake, Koehn, and Mertz 1999). This dataset captures financial and demographic information (including account information, credit history, employment, gender) of about 1000 loan applicants. The sensitive attribute in this dataset is gender – 69% of the individuals are male. Each individual in the dataset is labeled either as a good customer or a bad one depending on their credit risk.

**Experimental Setup** Here, we discuss the implementation details of our framework. We randomly split all the datasets into training (90%) and test (10%) sets. We build our framework using the training set and evaluate it on the test set.

*Biased Classifier  $f$ :* We construct  $f$  to be a perfectly discriminatory classifier that uses a sensitive feature i.e.,  $f$  makes predictions purely based on race in the case of COMPAS and Communities and Crime datasets (if  $race = African\ American$ , then predict 1, otherwise set predict 0), and based on gender for the German Credit dataset.

*Perturbations & OOD classifier:* For LIME, we generate a perturbed instance for every data point in the data set  $\mathcal{X}$  by adding random noise sampled from  $\mathcal{N}(0, 1)$  to each feature value. For SHAP, we randomly choose a subset of features for each data point and mark their values as missing by replacing them with their corresponding values from background distribution<sup>1</sup>. We choose the background distribution for SHAP to be cluster centers from kmeans with 10 clusters. We use these newly generated perturbations as well as the input data points to train the OOD classifier to detect OOD samples (See Section on Our Proposed Framework). We use the standard sklearn implementation of a random forest with 100 tree estimators as our OOD classifier.

*Unbiased Classifier  $\psi$ :* We build the unbiased classifier  $\psi$  by

<sup>1</sup>See Lundberg and Lee (2017) for more details.

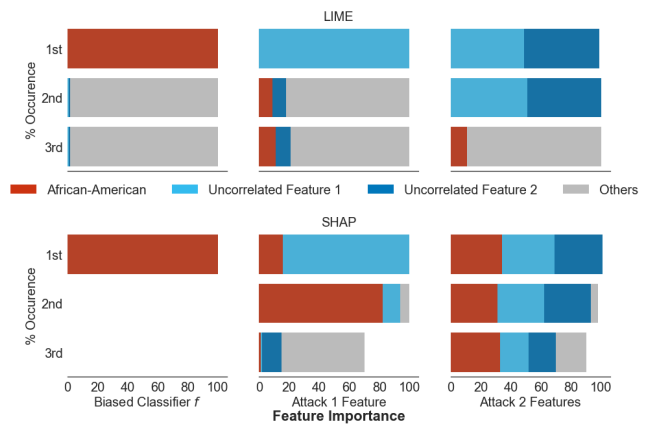


Figure 2: **COMPAS**: % of data points for which each feature (color coded) shows up in top 3 (according to LIME and SHAP’s ranking of feature attributions) for the biased classifier  $f$  (left), our adversarial classifier where  $\psi$  uses only one uncorrelated feature to make predictions (middle), and our adversarial classifier where  $\psi$  uses two uncorrelated features to make predictions (right).

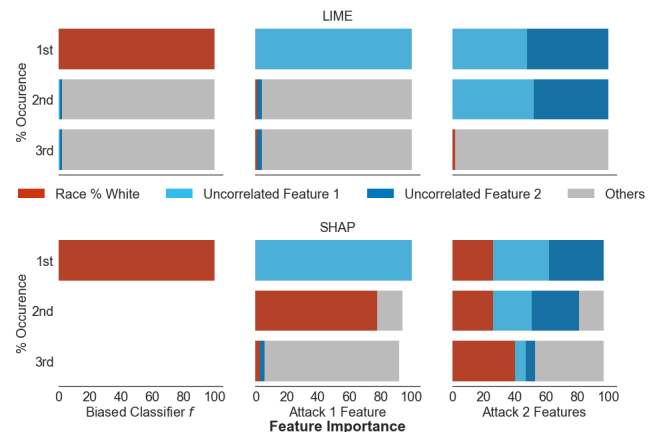


Figure 3: **Communities and Crime**: Similar to Fig 2, with *Race % White* as the sensitive feature.

constructing synthetic *uncorrelated features* which have zero correlation with sensitive attributes (e.g., race or gender). We experiment with one or two uncorrelated features. When we only have one uncorrelated feature in a particular experiment,  $\psi$  solely uses that to make predictions (if uncorrelated feature = 1, then predict 1, else predict 0). On the other hand, when we have two uncorrelated features in an experiment, we base the predictions on the xor of those two features. Note that  $\psi$  does not have to be restricted to always use synthetic uncorrelated features. It can also use any other existing feature in the data to make predictions. We experiment with synthetic uncorrelated features on COMPAS and CC dataset, and with *Loan Rate % Income* feature on the German credit dataset.

*Generating Post hoc Explanations:* We use the default LIME tabular implementation without discretization, and use the default Kernel SHAP implementation with background distribution set to kmeans with 10 clusters.

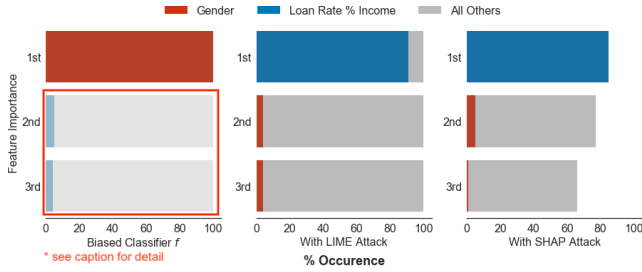


Figure 4: **German credit:** Similar to Fig 2 and 3, but  $\psi$  uses an existing feature (*Loan Rate % Income*) to make predictions, and *Gender* is the sensitive feature. Results for LIME are shown here.

### Evaluating Effectiveness of Adversarial Classifiers

To evaluate how successful our *attacks* are on LIME and SHAP, we compute the percentage of data points for which race, uncorrelated features (in case of COMPAS and CC) or *Loan Rate % Income* (in case of German credit data) show up in top 3 when features are ranked based on feature attributions output by LIME and SHAP. Results are shown in Figures 2, 3, and 4

In case of **LIME**, when a single feature is used for the attack i.e.,  $\psi$  uses a single feature for making predictions, the adversarial classifier  $e$  successfully shifts the feature importance in LIME from the true informative feature (the sensitive feature used by  $f$ ) to the uncorrelated feature in 100% of held-out points for COMPAS (Figure 2 - top and middle) and CC (Figure 3 - top and middle), and to the *Loan % Income* feature in 91% for German (Figure 4 - middle). Furthermore, the sensitive feature appears at 2nd or 3rd rank position roughly only as often as other features. When both uncorrelated features are used to define  $\psi$  (for COMPAS and CC), the 1st and 2nd rank positions for nearly all the data points are occupied by these features, with the sensitive feature entering into 3rd place at about the same rate as other features (Figures 2 & 3 - top right). These results demonstrate that the LIME explanation technique has been effectively fooled by the adversarial classifier  $e$  for all three datasets.

For **SHAP**, when a single uncorrelated feature is used for the attack, the adversarial classifier  $e$  successfully shifts the feature importance from the sensitive feature in 100% of data points in CC (Figure 3 - bottom and middle), 84% of data points in COMPAS (Figure 2 - bottom and middle), and 85% of data points in German. When two uncorrelated features are used in the attack (for COMPAS and CC), the adversarial classifier is less successful in removing the bias feature from 1st place in the ranking, succeeding in only 67% and 71% instances for COMPAS and CC respectively (Figures 2 & 3 - bottom right). This is due to SHAP’s local accuracy property that ensures that feature attributions must add up to the difference between a given prediction and the average prediction for the background distribution. This property will tend to distribute feature attributions across several features when it is not possible to identify a single most informative feature.

To ensure that the adversarial classifiers  $e$  are faithfully approximating the original classifiers  $f$  on instances from the original dataset, we evaluate the *fidelity* of  $e$  w.r.t.  $f$  by com-

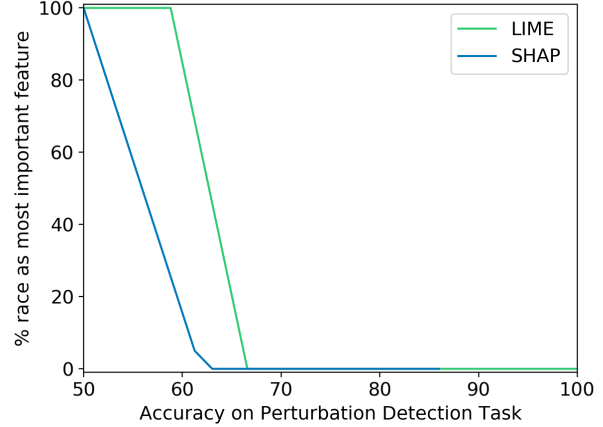


Figure 5: Effectiveness of our attack: % of data points for which race is the most important feature vs. the accuracy of the OOD classifier (perturbation detection) for both LIME and SHAP on the COMPAS dataset (averaged over three runs).

puting the percentage of held-out instances on which their predictions agree. In case of LIME, our adversarial classifiers perfectly reproduce the classifier  $f$  on all datasets 100% of the time. On the other hand, in case of SHAP, our adversarial classifiers are slightly less effective: 75% for CC, 80% for German, and 91% for COMPAS. These results indicate that our approach has constructed adversarial classifiers  $e$  that successfully fool LIME and SHAP, making them look like the unbiased  $\psi$  classifiers, while behaving like the original biased classifiers  $f$  on input data.

### Assessing the Robustness of Adversarial Classifiers

Here, we investigate the impact of various parameters of our framework on the effectiveness of our attacks.

**Perturbation Detection** Our approach relies on the OOD classifier to detect whether a given instance is a result of perturbation or not. Here, we investigate how accurate does this classifier need to be for our attacks to be effective. We train a number of different OOD classifiers with varying accuracies on the COMPAS dataset. For each OOD classifier, we evaluate the effectiveness of the resulting attacks (measured as percentage of data points for which race is the most important feature in the post hoc explanations). The results are presented in figure 5. When the OOD classifiers are unable to adequately differentiate between perturbed instances and points from the input dataset, i.e. accuracy is 50%, both LIME and SHAP explanations detect race as the most important feature. However, once the OOD classifier is a bit more accurate, the adversarial attacks are successful, and both LIME and SHAP do not identify race as the most important feature. Between the two, LIME requires the OOD classifier to be slightly more accurate for the attack to be effective, but the difference is not substantial. These results indicate that our attacks can be applied as long as it is possible to differentiate between the perturbed instances and the input data points.



Feature	Before	After attack
Race	-0.42	0.0
Uncorrelated feature 1	0.0	0.42
Average of other features	0.0	0.0

Figure 6: Feature coefficients of LIME explanations for an instance from COMPAS, before and after an attack ( $\psi$  uses a single feature).

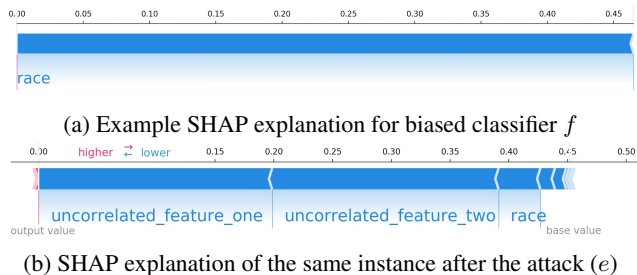


Figure 7: Examples of SHAP explanations for a single data point with biased classifier  $f$  (top) and our adversarial classifier  $e$  (bottom).

**Robustness to LIME/SHAP Hyperparameters** We analyze the behavior of our framework when different parameters are varied *at test time* when generating LIME and SHAP explanations for COMPAS; we summarize the key results here, with detailed results in the Appendix. First, we observe that classifiers that are adversarial to LIME are ineffective against SHAP explanations, while adversarial classifiers designed for SHAP have varying effectiveness against LIME. On varying the LIME kernel width between  $0.15\sqrt{M}$  to  $0.95\sqrt{M}$  (default kernel width is  $0.75\sqrt{M}$ , where  $M$  is the number of features), the attacks remain equally effective. Additionally, changing the LIME kernel to use  $l1$  norm for distance instead of the default  $l2$  norm also does not impact the effectiveness. For SHAP, we train our adversarial classifier assuming the default background distribution of K-means with 10 clusters; 5 and 15 clusters at test-time are also equally effective. Using a background distribution of all zeroes for SHAP makes our attacks even more pronounced.

### Example Explanations

Here, we present few example explanations from the COMPAS dataset. In case of LIME explanations in Table 6, we see that *Race* as a feature is completely hidden from the explanation after the attack, even though it was the sole important feature for the original classifier  $f$ . Similarly, for SHAP explanations in Fig 7, the sole important feature (*race*) is considerably hidden in the explanation after the attack, although not completely nullified as in the LIME explanation.

### Related Work

Below, we provide a brief overview of related work.

*Perturbation-based Explanation Methods:* Perturbation-based methods are a popular tool for post hoc feature attribution. In addition to LIME (Ribeiro, Singh, and Guestrin 2016) and SHAP (Lundberg and Lee 2017), a number of

other perturbation-based techniques have been proposed in literature. For instance, anchors (Ribeiro, Singh, and Guestrin 2018) use (non-linear) rules to express more actionable local explanations. GAM (Ibrahim et al. 2019) interprets local attributions as conjoined weighted rankings and uses k-medoids clustering to identify prototypical explanations.

*Criticism of Post hoc Explanations:* Rudin (2019) argues that post hoc explanations are not reliable, as these explanations are not necessarily faithful to the underlying models and present correlations rather than information about the original computation. Ghorbani, Abid, and Zou (2019) show that some explanation techniques can be highly sensitive to small perturbations in the input even though the underlying classifier’s predictions remain unchanged. Mittelstadt, Russell, and Wachter (2019) note that perturbation points created in LIME and SHAP are not at all intuitive, especially in case of structured data.

*Adversarial Explanations:* There has been some recent research on manipulating explanations in the context of image classification. Dombrowski et al. (2019) show that by modifying inputs in a way that is imperceptible to humans, they can arbitrarily change saliency maps. Heo, Joo, and Moon (2019) also propose similar attacks on saliency maps.

*Interpretability and Bias Detection:* Doshi-Velez and Kim (2017) argue that interpretability can help us evaluate if a model is biased or discriminatory. On the other hand, Lipton (2016) posits that post hoc explanations can never definitively prove or disprove unfairness of any given classifier. Selbst and Barocas (2018) and Kroll et al. (2016) provide examples showing that even if a model is completely transparent, it is hard to detect and prevent bias due to the existence of correlated variables.

### Conclusions and Future Work

We proposed a novel framework that can effectively hide discriminatory biases of any black box classifier. Our approach exploits the fact that post hoc explanation techniques such as LIME and SHAP are perturbation-based to create a *scaffolding* around the biased classifier such that its predictions on input data distribution remain biased, but its behavior on the perturbed data points is controlled to make the post hoc explanations look completely innocuous. Extensive experimentation with real world data from criminal justice and credit scoring domains demonstrates that our approach is effective at generating adversarial classifiers that can fool post hoc explanation techniques. We also found that LIME is more vulnerable to our *adversarial attacks* than SHAP. Our findings suggest that existing post hoc explanation techniques are not sufficiently robust for ascertaining discriminatory behavior of classifiers in sensitive applications.

This work paves way for several interesting future research directions in ML explainability. First, it would be interesting to systematically study if other classes of post hoc explanation techniques (e.g., gradient based approaches) are also vulnerable to adversarial attacks. Second, it would be interesting to develop new techniques for building *adversarially robust* explanations that can withstand attacks such as the ones outlined in this work.

## Acknowledgements

This work is supported in part by the Allen Institute for Artificial Intelligence (AI2) and in part by NSF award #IIS-1756023. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies

## References

- [2016] Angwin, J.; Larson, J.; Mattu, S.; and Kirchner, L. 2016. Machine bias. *ProPublica*.
- [2007] Asuncion, A., and Newman, D. 2007. Uci machine learning repository, 2007.
- [1999] Blake, C.; Koehn, E.; and Mertz, C. 1999. Repository of machine learning. *University of California at Irvine*.
- [2019] Dombrowski, A.-K.; Alber, M.; Anders, C. J.; Ackermann, M.; Müller, K.-R.; and Kessel, P. 2019. Explanations can be manipulated and geometry is to blame. *arXiv preprint arXiv:1906.07983*.
- [2017] Doshi-Velez, F., and Kim, B. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.
- [2019] Elshawi, R.; Al-Mallah, M. H.; and Sakr, S. 2019. On the interpretability of machine learning-based model for predicting hypertension. *BMC medical informatics and decision making* 19(1):146.
- [2019] Ghorbani, A.; Abid, A.; and Zou, J. 2019. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3681–3688.
- [2019] Heo, J.; Joo, S.; and Moon, T. 2019. Fooling neural network interpretations via adversarial model manipulation. *arXiv preprint arXiv:1902.02041*.
- [2019] Ibrahim, M.; Louie, M.; Modarres, C.; and Paisley, J. 2019. Global explanations of neural networks: Mapping the landscape of predictions. *arXiv preprint arXiv:1902.02384*.
- [2017] Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C.; Wexler, J.; Viegas, F.; and Sayres, R. 2017. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *arXiv preprint arXiv:1711.11279*.
- [2016] Kroll, J. A.; Barocas, S.; Felten, E. W.; Reidenberg, J. R.; Robinson, D. G.; and Yu, H. 2016. Accountable algorithms. *U. Pa. L. Rev.* 165:633.
- [2016] Larson, J.; Mattu, S.; Kirchner, L.; and Angwin, J. 2016. How we analyzed the compas recidivism algorithm. *ProPublica* (5 2016) 9.
- [2016] Lipton, Z. C. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.
- [2017] Lundberg, S. M., and Lee, S.-I. 2017. A unified approach to interpreting model predictions. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Neural Information Processing Systems (NIPS)*. Curran Associates, Inc. 4765–4774.
- [2019] Mittelstadt, B.; Russell, C.; and Wachter, S. 2019. Explaining explanations in ai. In *Proceedings of the conference on fairness, accountability, and transparency*, 279–288. ACM.
- [2002] Redmond, M., and Baveja, A. 2002. A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research* 141(3):660–678.
- [2011] Redmond, M. 2011. Communities and crime un-normalized data set. *UCI Machine Learning Repository*. In website: <http://www.ics.uci.edu/mllearn/MLRepository.html>.
- [2016] Regulation, G. D. P. 2016. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46. *Official Journal of the European Union (OJ)* 59(1-88):294.
- [2016] Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Knowledge Discovery and Data Mining (KDD)*.
- [2018] Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2018. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- [2019] Rudin, C. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1(5):206.
- [2018] Selbst, A. D., and Barocas, S. 2018. The intuitive appeal of explainable machines. *Fordham L. Rev.* 87:1085.
- [2018] Tan, S.; Caruana, R.; Hooker, G.; and Lou, Y. 2018. Distill-and-compare: auditing black-box models using transparent model distillation. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 303–310. ACM.
- [2016] Whitmore, L. S.; George, A.; and Hudson, C. M. 2016. Mapping chemical performance on molecular structures using locally interpretable explanations. *arXiv preprint arXiv:1611.07443*.

## Appendices

### A Details of the Algorithms

Symbol	Description
$x_i$	Observed attributes of a data point $i$
$y_i$	ground truth class label of a data point $i$
$\mathcal{X}$	Set of all the observed attributes of input data points i.e., $\mathcal{X} = \{x_1, x_2, \dots, x_N\}$
$\mathbf{y}$	Set of all the ground truth class labels of input data points i.e., $\mathbf{y} = \{y_1, y_2, \dots, y_N\}$
$\mathcal{D}$	Set of input data points $\mathcal{D} = (\mathcal{X}, \mathbf{y}) = \{(x_1, y_1), (x_2, y_2) \dots (x_N, y_N)\}$
$\mathcal{C}$	Set of class labels in $\mathcal{D}$ i.e., $\forall i, y_i \in \mathcal{C}$
$\mathcal{X}_{dist}$	Distribution from which $\mathcal{X}$ is sampled
$\mathcal{D}_{dist}$	Distribution from which $\mathcal{D}$ is sampled
$f$	Black box model which maps a data point to a class label i.e., $f(x_i) \in \mathcal{C}$
$g$	Interpretable model that serves as an explanation of the black box model $f$ generated by posthoc explanation techniques
$\psi$	Unbiased classification function
$e$	Adversarial classifier
$\mathcal{X}_p$	Set of perturbed data points generated from $\mathcal{X}$

Table 1: Description of Notation



---

**Algorithm 1** Building OOD Sample Detection Classifier

---

```
1: Input: Input data  $\mathcal{X}$ ; perturb();
2: Output: OOD Sample Detection Classifier
3:  $\mathcal{X}_p = \{\}$  ▷ Set of perturbed data points
4:
5: for  $x \in \mathcal{X}$  do
6:    $x_p = \text{perturb}(x)$ 
7:    $\mathcal{X}_p = \mathcal{X}_p \cup x_p$ 
8: end for
9:
10: for  $x \in \mathcal{X}$  do ▷ Input data points are not OOD
11:   assign label False
12: end for
13:
14: for  $x_p \in \mathcal{X}_p$  do ▷ Check if a perturbed point is OOD
15:   if  $\exists x \in \mathcal{X}, x_p \approx x$  then assign label False
16:   else assign label True
17:   end if
18: end for
19:
20: Train a classifier is_OOD on  $\mathcal{X} \cup \mathcal{X}_p$  and their corresponding labels (assigned above).
21:
22: return is_OOD
```

---

---

**Algorithm 2** Adversarial Classifier  $e$ 

---

```
1: Input: Biased Base classifier  $f$ ; Unbiased classifier  $\psi$ ;
2: Sample data point  $x$ ; is_OOD();
3: Output: Prediction of adversarial classifier  $e$  on input data point  $x$ 
4:
5: if not is_OOD( $x$ ) then  $e_x = f(x)$ 
6: else  $e_x = \psi(x)$ 
7: end if
8:
9: return  $e_x$ 
```

---

## B Summary of Datasets

Dataset	Size	Features	Positive Class	Sensitive Feature
COMPAS	6172	criminal history, jail and prison time, demographics, COMPAS risk score	High Risk (81.4%)	African-American (51.4%)
Communities and Crime	1994	race, age, education, marriage status, citizenship, police demographics	Violent Crime Rate (50%)	White Population (continuous)
German Credit	1000	account information, credit history, loan purpose, employment, demographics	Good Customer (70%)	Male (69%)

Table 2: Dataset information and the proportion of data that belongs to the positive class or the sensitive feature

## C Detailed Results of Effectiveness Evaluation

COMPAS LIME Adversarial Classifier							
	Baseline classifier $f$	Attack 1 feature			Attack 2 features		
Importance Ranking	1	1	2	3	1	2	3
African-American	100	0	9	11	0	0	11
Unrelated Feature 1	0	100	0	0	49	51	0
Unrelated Feature 2	0	0	9	10	50	49	0
Other Features	0	0	82	79	0	0	89
Accuracy	56	56			56		

COMPAS SHAP Adversarial Classifier							
	Baseline classifier $f$	Attack 1 feature			Attack 2 features		
Importance Ranking	1	1	2	3	1	2	3
African-American	100	16	82	1	34	31	33
Unrelated Feature 1	0	84	12	1	35	31	19
Unrelated Feature 2	0	0	0	13	32	31	18
Other Features	0	0	6	55	0	5	20
Accuracy	56	53			55		

Table 3: The percentage occurrence of the top features in  $e$  with the perfectly unfair  $f$  (if African-American predict high likely to recidivate) on COMPAS LIME and SHAP adversarial explanations. “Attack 1 feature” and “Attack 2 features” can be read as two different scenarios: the former where one feature is used in  $\psi$  and the latter where two features are used in  $\psi$ . The baseline is the explanation on  $f$ . The percentages are the mean rates of occurrences on a 10% test set over three different runs (including re splitting into a different train-test set). We see that whether the individual is African American is the most important feature in both LIME and SHAP on  $f$ . Using  $e$ , we are able to exclude all of the explanations of this feature in LIME and most of the explanations in SHAP in the top feature positions.

<b>Communities and Crime LIME Adversarial Classifier</b>							
	Baseline classifier $f$	Attack 1 feature			Attack 2 features		
Importance Ranking	1	1	2	3	1	2	3
Race % White	100	0	1	1	0	0	2
Unrelated Feature 1	0	100	0	0	48	52	0
Unrelated Feature 2	0	0	3	3	52	48	0
Other Features	0	0	96	96	0	0	98
Accuracy	73	73			73		

<b>Communities and Crime SHAP Adversarial Classifier</b>							
	Baseline classifier $f$	Attack 1 feature			Attack 2 features		
Importance Ranking	1	1	2	3	1	2	3
Race % White	100	0	78	3	26	26	40
Unrelated Feature 1	0	100	0	0	36	25	7
Unrelated Feature 2	0	0	0	3	35	30	6
Other Features	0	0	16	86	0	16	44
Accuracy	73	70			72		

Table 4: The percentage occurrence of the top features in  $e$  with the perfectly unfair  $f$  (if race % white > median race % white predict nonviolent community) on Communities and Crime. Using the  $e$ , we are able to exclude all of the explanations of this feature in LIME and many of the SHAP explanations, consistent with our results on COMPAS.

<b>German Credit LIME Adversarial Classifier</b>				
	Baseline classifier $f$	With Attack		
Importance Ranking	1	1	2	3
Gender	100	0	4	4
Loan Rate % Income	0	91	0	0
Other Features	0	9	96	96
Accuracy	64	64		

<b>German Credit SHAP Adversarial Classifier</b>				
	Baseline classifier $f$	With Attack		
Importance Ranking	1	1	2	3
Gender	100	0	5	1
Loan Rate % Income	0	85	0	0
Other Features	0	0	72	65
Accuracy	64	64		

Table 5: The percentage occurrence of the top features in  $e$  with the perfectly unfair  $f$  (if Gender is male predict will repay loan) on COMPAS LIME and SHAP explanations. We use loan rate as a percentage of income as  $\psi$  and predict false if the value is above its mean. In both the LIME and SHAP case, we are able to exclude gender from the majority of the explanations. When the explanation is included, it appears at the same frequency as other features.

## D Detailed Results of the LIME/SHAP Hyperparameters

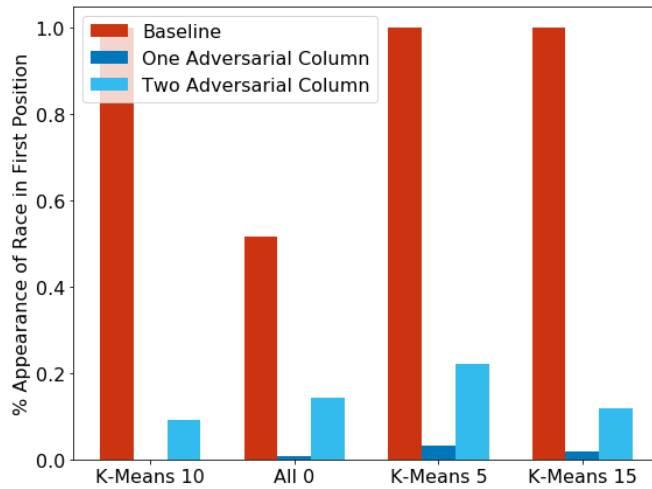


Figure 8: SHAP attack effectiveness across different background distributions: K-means 10 is the distribution assumed in training. We also test on K-means 5, K-means 15, and all 0. These represent different suggestions in the SHAP software package for representing a large dataset.

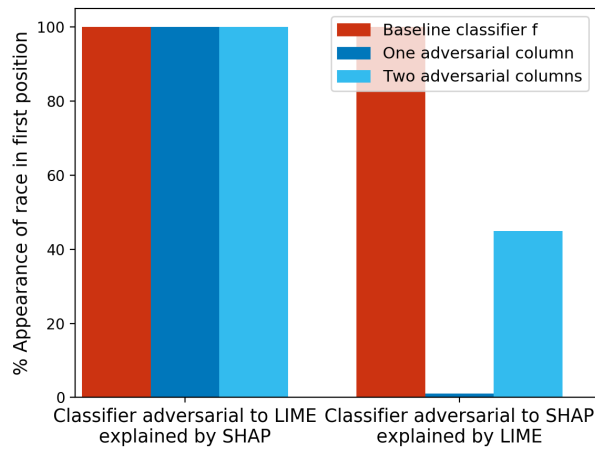


Figure 9: The effectiveness of explaining classifiers adversarial to SHAP with LIME and classifiers adversarial to LIME with SHAP. We see that classifiers adversarial to LIME are not effective at hiding bias from SHAP explanations, and classifiers adversarial to SHAP are moderately effective at hiding biases from LIME explanations.