Don't Lie to Me: Avoiding Malicious Explanations with STEALTH

Lauren Alvarez, Tim Menzies,
North Carolina State University, USA
lalvare@ncsu.edu, timm@ieee.org,
Dec 30, 2022

Abstract—**Before we can trust AI models, we need some way to assess them without being misled by malicious algorithms. *STEALTH* avoids such algorithms by building models and explanations from a tiny sample of the data. Such sparse sampling is so undetectable that malicious algorithms never know when to lie. Another reason to use *STEALTH* is that the models generated this way actually perform as well as or better than models generated from all the data.**

In order to support open science practices, all our scripts and data are on-line at https://github.com/laurensalvarez/STEALTH.

Introduction

Within the current AI industry there are many available models– not all of which can be inspected. "Model stores" are cloud-based services that charge a fee for using models hidden away behind a firewall (e.g. AWS market-place [1] and the Wolfram neural net repository [2]). Adams et al. [3] discusses model stores (also known as "machine learning as a service" [4]), and warns that these models are often low quality (e.g. if it comes from a hastily constructed prototype from a Github repository, dropped into a container, and then sold as a cloud-based service). Ideally, we use software testing to defend ourselves against potentially low quality models. But model owners may not publish verification results or detailed specifications– which means standard testing methods are unsure what to test for.

An alternative to standard testing is to run an explanation algorithm that offers a high-level picture of how model features influence each other. Unfortunately, the better we get at generating explanations, the better we also get at generating misleading explanations. As shown below:

> The more a model is queried, the better it can lie.

For example, Slack et al.'s lying algorithm [8] (discussed below) knows how to detect "explanation-oriented" queries. That liar algorithm can then switch to models which, by design, disguise biases against marginalized groups (e.g. some specific gender, race, or age grouping). The Slack et al.'s results are particularly troubling. An alarming number commercially deployed models having discriminatory properties [5], [6]. For example, the (in)famous COMPAS model (described in Table **??**) decides the likelihood of a criminal defendant reoffending. The model suffers from alarmingly different false positive rates for Black defendants than White defendants. Noble's book *Algorithms of Oppression* offers a long list of other models with discriminatory properties [5].

Clearly, before we can trust models from the cloud, we need some way to assess their bias without being mislead by malicious algorithms. This article reasons as follows. If too many queries are the problem, then perhaps the solution is:

> Ask fewer queries.

To test this idea, we built a new algorithm called *STEALTH* which recursively bi-clusters N examples, down to leaves of size $\sqrt{N}$. A model is then queried with 2 examples per leaf. These queries generate "labels"; i.e. decisions about each example. These $2\sqrt{N}$ examples are used to build a *surrogate* model which can be used for predictions and explanations. The key point here is that since *STEALTH* makes few queries, malicious algorithms can not detect its operation. Hence, they never know when to lie. Also, since we use our own surrogate for explanations, those explanations cannot be distorted by a malicious model.

But this all assumes the surrogate (learned from very few samples) can mimic the important properties of the original model (learned from much more data). To test that, our research questions ask what is *gained* and *lost* by reasoning over such a small $2\sqrt{N}$ sample of the data:

RQ1: *Does our method prevent lying?* When applying Slack et al.'s lying algorithm, we found while the original model can lie, *lying fails in our surrogates*.

RQ2: *Does the surrogate model perform as well as the original model?* Much to our surprise, in the usual case, STEALTH*'s surrogates performed as well or better as the original* measured in terms of *both* predictive performance *and* bias mitigation.

We will also see that *STEALTH* is competitive to other state-of-the-art bias reduction methods such as Fair-Smote [6], MAAT [9], and FairMASK [11]. Since these other methods have no defenses against malicious algorithms, we recommend *STEALTH* over the rest.

Background (How to Lie)

Figure 1.A shows the Local Interpretable Model-Agnostic Explanations (LIME) algorithm [7] that samples instances uniformly at random near an example point, then builds a class of linear models from the generated sample data. Using its models, LIME can explain what features influence the classification of a particular instance.

LIME's explanations looks like Figure 1.B. Note that, in the figure, the Age feature is most influential. By running LIME on all the test instances, it is possible to collect the *most influential* features; i.e. the attributes that are found as being most influential for the test set.

An interesting result seen in the most influential feature set is that often most features are *not* influential. For example, in our study, we have ten data sets with dozens of attributes, but only a handful are ranked most influential. This indicates the most influential features are a useful way to summarize the differences between explanations generated via different methods. Given two most influential sets A, B a Jaccard coefficient

Figure 1.A: LIME samples around the boundary to finds the delta between blue and red classes. From [7].

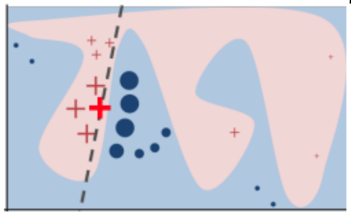

Figure 1.B: Heart Feature importance as assessed by LIME. A **positive** weight means the feature encourages the classifier to predict the instance as a positive and vice versa for the **negative** weight. Larger weights indicate greater feature importance.

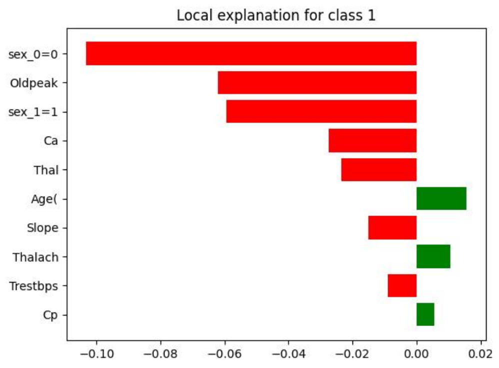

Figure 1.C: Blue/red = original/invented data. From [8].

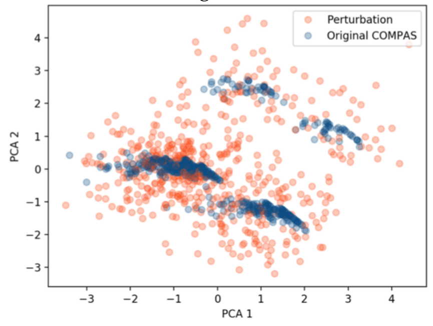

Fig. 1. About LIME.

is computed between (0,1) meaning (zero,total) similarities in the explanations, respectively:

$$jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

To build a lying algorithm, we note that, in Figure 1.C, many of the **red** dots (showing LIME's generated samples) are at different locations to the raw data (see the **blue** dots). Slack et al. [8] built a classifier to recognize when many of these **red** dots appear as input to the model. This meant they could distinguish (a) queries-about-explanation from (b) normal-queries seen in day-to-day operation. Hence they could hide any discriminatory properties by offering a *lying model* to queries-about-explanation and a second model to normal-queries.

To build their liar, Slack et al. trained an out-of-distribution (OOD) model using similar mutations to LIME, and used this OOD detector to produce different classifications using the *protected attributes*. These are special attributes that divide data into two, *privileged* & *unprivileged*, groups (e.g. such as *male* & *female*; *young* & *old*). Thus, by training a model to recognize OOD data and in-distribution data, it is possible to hide model discriminatory classifications against *unprivileged groups*.

The problems raised by Slack et al. [8] are widely explored–see Table I. In that table, *STEALTH* use of limited, local data as samples is novel. But do such small samples of local data exist? Quite possibly. We claim, without controversy, that for a model to sell at a model store *someone must want to use it*. That is, model store users must know their own domain data that they wish to label. For example, a doctor could have many records of patients, but wants to query a model to obtain a label of "high risk" or "low risk" of an imminent heart attack.

Using that local data, a very small number of local domain examples could be passed to the model store, from which we could build a *surrogate* model. If the number of queries was kept low enough, then such a *STEALTH* approach would be inherently undetectable since:

- It would add very few **red** points to Figure 1.
- The points it adds would be very close to the **blue** dots (since they are local data samples)

Methods

Table II describes the seven steps needed to run and evaluate *STEALTH*. To answer our research questions, we applied those steps, with and without Slack et al.'s lying method [8]. The models built in this way were assessed via their (a) explanation properties (using Equation 1); (b) their predictive performance; and (b) their fairness using metrics that are widely applied in the literature (see Table III). Since we are exploring fairness, we also compare *STEALTH*'s to state-of-the-art bias reduction algorithms (Fair-Smote [6], MAAT [9], and FairMASK [11].

ALGORITHMS: For this work, we tried various learners such as random forests, logistic regression, and a SVM with radial basis function. It was observed that random forests generated better recalls than anything else. Hence, we focus on that learner for this work.

Table II uses several bias reduction methods. Chen et al.'s MAAT system [9] makes conclusions by balancing conclusions between two separate models (a performance model and a fairness. model). Chakraborty et al.'s Fair-SMOTE system [6] adjusts the training data by (a) removing training instances that change conclusions when protected attributes change; and (b) evening out the distributions of the protected attribute values. Chakraborty argues that these adjustments make it harder for any particular protected value to unduly effect the conclusion. Peng et al.'s FairMASK system [11] replaces protected attributes with new values learned from other independent attributes. Peng argues that this removed misleading latent correlations between attributes since the learned values tend to be average values (not outliers).

An important note to make about MAAT, Fair-SMOTE and FairMASK is that all these methods require the analyst to pre-specify the protected attributes. *STEALTH*, on the other hand, operates without such knowledge. Despite this, as we will see, *STEALTH* achieves competitive levels of bias reduction to MAAT, Fair-SMOTE and FairMASK.

DATA: The case studies in Table **??** were selected since they appeared in prior explanation and fairness papers including Slack et al. [8]. In column three of Table **??**, each protected attribute is listed.

STATISTICS: Eight of our data sets have one protected attribute while ADULT and COMPAS have two. Hence, we run 12 times, one for each data set and protected attribute. Each run is repeated 20 times, each time dividing the original data

TABLE I
RELATED WORK

If the problem is that, in Figure 1.C, the **red** dots are different to the **blue** then one solution is to make LIME's perturbations more realistic by (e.g.) adjusting sampling distributions [1], or better neighborhood calculations [2,3,4,5]. Other approaches improve LIME by making it more robust and less vulnerable to attacks [4,5], or creating better detection defenses [6,7] This is a very active area of research. For example, see [8-12] for papers where prior work is rapidly assessed and improved/refuted by subsequent work.

We have three comments on that related work. Firstly, some of that work makes restrictive assumptions about the data. For example, Ji et al. [1] rely on parametric assumptions such as the the data and model output conform to a normal distribution. While such assumptions might be true, they cannot be checked in the black-box case. *STEALTH*, on the other hand is a non-parametric instance-based approach that makes no assumptions of (e.g.) normality.

Secondly, some of the above are very slow. GAN (generative adversarial networks) is a technique to generate data that conforms to some unknown distribution [4]. That approach requires the execution of multiple rounds of two deep learners (one to generate the data and the other to classify generated data as within, our outside, of the real distribution). *STEALTH* on the other hand, is much faster; specifically, the median to max runtimes for our approach on the data explored here is just 11 seconds to 5 minutes.

Thirdly, the above work does not take full advantage of naturally occurring data. Users of models in a model store must have a supply of their own domain data which they wish to label. As discussed in the main text, we can use that to great effect.

Related work citations:

1) D. Ji, P. Smyth, and M. Steyvers, "Can i trust my fairness metric? assessing fairness with unlabeled data and bayesian inference," in NeurIPS'20.
2) D. Vreˇs and M. R. ˇSikonja, "Better sampling in explanation methods can prevent dieselgate-like deception," arXiv:2101.11702, 2021.
3) Y. Jia, J. Bailey, et al. "Improving the quality of explanations with local embedding perturbations," in KDD 2019, pp. 875–884.
4) S. Saito, E. Chua, N. Capel, and R. Hu, "Improving lime robustness with smarter locality sampling," arXiv preprint arXiv:2006.12302, 2020.
5) A. Saini and R. Prasad, "Select wisely and explain: Active learning and probabilistic local post-hoc explainability," AAAI/ACM Conference on AI, Ethics, and Society, 2022, pp. 599–608.
6) Z. Carmichael and W. J. Scheirer, "Unfooling perturbation-based post hoc explainers," arXiv preprint arXiv:2205.14772, 2022.
7) J. Schneider, J. Handali, M. Vlachos, and C. Meske, "Deceptive ai explanations: Creation and detection," arXiv preprint arXiv:2001.07641, 2020
8) Slack, D., Hilgard, A., Lakkaraju, H., & Singh, S., "Counterfactual explanations can be manipulated," in NeurIPS '21
9) Wilking, R., Jakobs, M., & Morik, K., "Fooling Perturbation-Based Explainability Methods," In ECML/PKDD '22.
10) Molnar, C., König, G., Herbinger, J., Freiesleben, T., Dandl, S., Scholbeck, C. A., ... & Bischl, B., "General pitfalls of model-agnostic interpretation methods for machine learning models,". xxAI - Beyond Explainable AI. 2020.
11) Maratea, A., & Ferone, A., "Pitfalls of local explainability in complex black-box models," In WILF. 2021.
12) Covert, I., et al. "Explaining by Removing: A Unified Framework for Model Explanation," J. Mach. Learn. Res. 22 (2021): 209-1.

1. Baseline generation: Build *MODEL1* (also known as the *original model*) using the 80% training data. In terms of our domain modelling, MODEL1 is the model hiding behind a firewall. Using MODEL1 and the 20% hold-out test data, we can collect *baseline values* for all the metrics of Table III.

2. Clustering: A *clustering algorithm* recursively bi-cluster the training data to leaves of size $\sqrt{N}$. We use Figure 2's recursive bi-clustering method. Nair et al. report that this method is fast and is useful for finding a good spread of examples across data [10].

3. Sampling: After that, we need a *leaf sampling algorithm* to select m examples per leaf. Here, we selected $m = 2$ items at random. While *STEALTH* seems to work at $m = 1$, other algorithms such as MAAT [9] seem to fail for such small data sets. Hence we use $m = 2$ since if go any smaller, we cannot validate them with respect to prior work.

4. Labelling: A *labelling algorithm* must then assign labels to $2\sqrt{N}$ selected examples. For that purpose, we query MODEL1.

5. Surrogate Creation: Some *model creation algorithm* must build a surrogate model. Here again, we used a random forest to build *MODEL2* (also known as the *surrogate model*) using the data associated with the $2\sqrt{N}$ labels.

6. Collect Performance Scores: Here, the surrogate MODEL2 is applied to the 20% hold-out test data to generate the Table III performance scores. These values are compared to the *baseline values* seen in **step 1.**

7. Collect Explanation Scores: Here, we gather the most influential attributes of (b1) MODEL1 and (b2) MODEL2 using LIME, and compare them using Equation 1.

TABLE II
THE SEVEN STEPS OF *STEALTH*. THE CORE OF *STEALTH* IS STEPS 2,3,4,5. STEPS 1,6,7 ARE ADDED FOR EVALUATION. THE EXPERIMENTS OF THIS PAPER TESTED THIS RIG 20 TIMES FOR EACH DATA SET OUR STUDY. NOTE THAT FOR EACH RUN, THE DATA IS DIVIDED 80:20 INTO A TRAIN:TEST SET. THE TRAINING DATA IS USED IN *all* THE FOLLOWING STEPS. THE TEST SET IS USED AS A SET OF HOLD-OUT EXAMPLES IN STEPS 1, 6, AND 7.

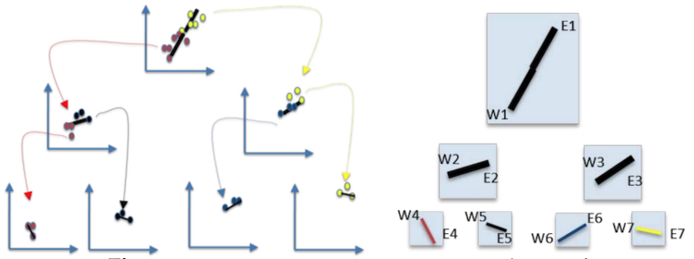

Figure 2a. **Figure 2b.**

NOTES: This recursive bi-clustering methods finds two distant points E (east) and W (west). Using $c = dist(E, W)$ then all other examples have distances a, b to E, W, respectively and distance $x = (a^2 + c^2 - b^2)/(2c)$ on a line from E to W. By splitting data on median x, the examples can be then bi-clustered (and so-on, recursively, see Figure 2b).

Fig. 2. *STEALTH* partitions x values using the random projections. Nair et al. report that this method is fast and is useful for generating a good spread of examples across a data set [10].

Performance Metric	Fairness Metric
Accuracy = (TP+TN)/(TP+TN+FP+FN)	**Average Odds Difference (AOD)**: Average of difference in False Positive Rates(FPR) and True Positive Rates(TPR) for unprivileged and privileged groups. $AOD = ((FPR_U\text{-}FPR_P) + (TPR_U\text{-}TPR_P))/\ 2$
False alarm = FP/(FP+TN)	**Equal Opportunity Difference (EOD)**: Difference of True Positive Rates(TPR) for unprivileged and privileged groups. $EOD = TPR_U - TPR_P$
Recall = TP/(TP+FN)	**Statisticl Parity Difference (SPD)**: Difference between probability of unprivileged group (protected attribute PA = 0) gets favorable prediction ($\hat{Y} = 1$) & probability of privileged group (protected attribute PA = 1) gets favorable prediction ($\hat{Y} = 1$). $SPD = P[\hat{Y} = 1 \vert PA = 0] - P[\hat{Y} = 1 \vert PA = 1]$
Precision = TP/(TP+FP)	**Disparate Impact (DI)**: Similar to SPD but instead of the difference of probabilities, the ratio is measured. $DI = P[\hat{Y} = 1 \vert PA = 0]/P[\hat{Y} = 1 \vert PA = 1]$
F1 Score = 2 × (Precision × Recall)/(Precision + Recall)	

TABLE III
EVALUATION METRICS. IN THIS TABLE TP, TN, FP, FN ARE THE TRUE/FALSE POSITIVE/NEGATIVE RATES SEEN IN BINARY CLASSIFICATION. THE DEFINITION FOR TRUE POSITIVE RATES (TPR) IS TP/(TP + FN) AND FALSE POSITIVE RATES (FPR) IS FP/(FP + TN).

into 80% training and 20% test. Since each run returns nine scores (one for each of the metrics of Table III), the generates 12*20*9=2160 scores.

For each data set and metric, we compared methods (using *STEALTH* or MAAT or FairSMOTE or FairMASK) against a **baseline** (running all data through random forests). Each method earns one more win or tie or loss if it is statistically better, same, or worse than the **baseline** (respectively). For that statistical test, we use a nonparametric tests recommended by prior work on bias reduction; i.e. a Scott-Knot procedure using Cliffs Delta and a bootstrap test. For notes justifying and explaining those procedures, see [12], [13], [6], [9]. Note that for recall, precision, accuracy and F1, *larger* numbers are *better* while for all other metrics such as false alarms, *smaller* numbers are *better*.

RESULTS

RQ1: *Does our method prevent lying?*

Table V compares the explanations from Slack et al.'s lying algorithm and *STEALTH*'s Slack surrogate models. In Table V, we see that the Slack et al. entries are nearly all zero; i.e. explanations from *STEALTH* do not overlap with what the lying algorithm's explanations. This means that Slack's algorithm cannot detect the operation of *STEALTH*, and so it never knows when to lie.

Generating different explanations (from the liar) is all very well, but that could be achieved just by selecting random values. Happily, we can show that the explanations from *STEALTH* correspond to important influences within our models. Table V also shows us that explanations from *STEALTH* have a large overlap (median=71.5%) with the explanations from the original model.

In summary to answer to RQ1, we say *STEALTH* can defeat Slack et al.'s lying algorithm. This is not to say that *STEALTH* defeats all possible liars (and exploring our method in the context of other liars is a clear direction for future work).

While RQ1 is a positive result (for our methods), many of the explanation overlap reported in Table V are less than 100%. That is, between the original model and those made by *STEALTH*, there there are differences (measured in terms of which attributes are most influential). Happily, the results of RQ2 suggest that *STEALTH*'s models are actually preferable to the original (which means it could be argued that we should have more confidence in the most influential attributes, as reported by *STEALTH*).

RQ2: *Does the surrogate model perform as well as the original model?*

TABLE V
RQ1 RESULTS: JACCARD COEFFICIENT RESULTS COMPARING THE FIRST TOP-RANKED FEATURES BY LIME OF THE ORIGINAL MODELS (USING THE BASELINE AND THE SLACK ET AL. LIAR ALGORITHM) TO *STEALTH*'S FIRST TOP-RANKED FEATURES. CALCULATED USING EQUATION 1.

Data set	Slack_Jacc	Base_Jacc
student	0	0.30
communities	0	0.44
heart	0	0.62
compas_r	0	0.62
meps	0	0.64
compas_s	0	0.68
		0.71.5 $\Longleftarrow$ median
bank	0.06	0.75
default	0.07	0.75
adult_r	0	0.75
adult_s	0	0.75
german	0	0.77
diabetes	0	0.88

It turns out that our our method for avoiding liars does not lead to sub-optimal models. Table VI shows how often our method finds improvements over the **baseline** models. Note that there are many wins recorded in Table VI; i.e. our method achieves numerous non-small, statistically significant improvements over that baseline (measured in terms of performance and fairness).

There are some losses reported in Table VI and to explain those, we look at the raw values of the **baseline** models (shown in Figure 3). It is clear that in the original model, there are some weakly performing data sets. For example, in Figure 3, while ADULT has high accuracies, the baseline also suffers from very high false alarms (over 50%). This is relevant since, when we drill down into the losses of Table VI, we note that many of *STEALTH*'s losses arise in ADULT. That is to say, these losses seem to be less about drawbacks with *STEALTH* and rather drawbacks with random forests (for these data sets).

In summary, to answer RQ2, we see no great loss from building surrogates from small original data samples. Further, we can also see in Table VI that *STEALTH* is competitive with state-of-the-art bias mitigation algorithms because it has very similar wins+ties for both performance and fairness metrics as the other methods:

- A case might be made that Fair-SMOTE performs a little better since it has more wins+ties for fairness;
- A similar case might be made for the superiority of MAAT since it has most fairness wins.

That said:

- The Table VI *STEALTH* values are close to Fair-SMOTE & MAAT (with FairMASK coming in fourth place).

	Performance: (accuracy, recall, precision, F1, false alarm)				Fairness: (AOD, EOD, SPD, DI)			
Method	Wins	Loses	Ties	Wins + Ties	Wins	Loses	Ties	Wins + Ties
STEALTH	29	16	15	44 / 60	20	11	17	37 / 48
MAAT	34	17	9	43 / 60	31	7	10	41 / 48
Fair-SMOTE	27	17	16	43 / 60	28	6	14	42 / 48
FairMask	21	23	16	37 / 60	12	25	11	23 / 48

TABLE VI
RQ2 RESULTS: A METHOD EARNS ONE WIN OR TIE OR LOSS IF IT IS STATISTICALLY **better**, SAME, OR WORSE (RESPECTIVELY) THAN THE FIGURE 3 BASELINE (AS ASSESSED VIA SCOTT-KNOT STATISTICAL TESTS THAT COMBINE CLIFFS DELTA AND A BOOTSTRAP). NOTE THAT FOR RECALL, PRECISION, ACCURACY AND F1, *larger* NUMBERS ARE *better* WHILE FOR ALL OTHER METRICS SUCH AS FALSE ALARMS, *smaller* NUMBERS ARE *better*.

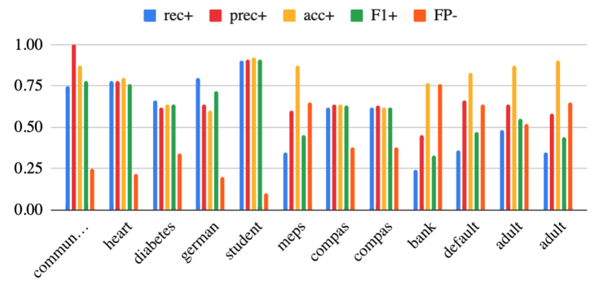

Figure 3.A: RQ2 results: *performance* reports median results for accuracy, recall, false alarm precision, F1.

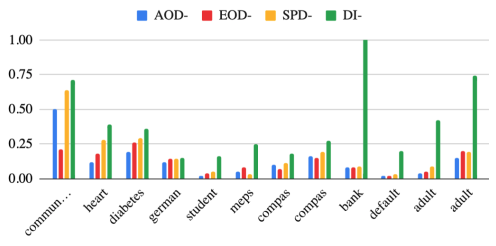

Figure 3.B: RQ2 results: *fairness* reports median results for EOD, AOD, SPD, DI

Fig. 3. RQ2 **Baseline** results on the Table III metrics.

- Fair-SMOTE, MAAT, and FairMASK have no defense against the Slack et al. lying algorithm.
- Also, other bias reduction methods have to be directed to protect specific attributes. *STEALTH*, on the other hand, offers the bias improvements without having to be directed. This is to say *STEALTH* could protect important attributes even if they were inadvertently overlooked.

DISCUSSION: WHY DOES IT WORK SO WELL?

By using few examples, *STEALTH* runs “under the radar” and is undetected by Slack et al.'s lying algorithm. *STEALTH* makes very few ($2\sqrt{N}$) in-distribution queries, which is undetectable by OOD attacks. Since *STEALTH*'s queries are undetectable, the liar cannot lie and returns its honest results.

Why do models built from just $2\sqrt{N}$ examples have similar or better performance, and bias reduction properties? Perhaps this is for the same reason that semi-supervised reasoning [14] works so well. Semi-supervised learners assume “a manifold assumption” i.e. data can be approximated by a smaller set of most important attributes, without much loss of signal [14]. As evidence of our data having a lower dimensional manifold, recall from the above that in all the data sets used here, there were very few most influential attributes. When such smaller manifolds exist, data mining needs to only explore a small sample of the data set (provided the samples are spread across the data set using methods like, for example, Figure 2).

As to *STEALTH*'s success at reducing bias, we conjecture that *STEALTH*'s success results from a zealous use of design principles from Peng et al.'s FairMASK [11]. Peng et al. argue that FairMASK reduces bias by replacing minority outlier opinions with values that are better supported by more local data. FairMASK does this by synthesizing replacement values for protected attributes from the other independent attributes. *STEALTH*, on the other hand, does this for *all independent attributes* since, in effect, its $2\sqrt{N}$ examples are reports of average case effects within small data clusters. By averaging the data in this way, we *smooth* out the features including any outliers that may skew results unfairly.

Based on the above, *STEALTH* is recommended over MAAT, Fair-SMOTE and FairMASK. But more generally, we argue that there is another result here. An important feature of these results is that they suggest a new view on the nature of explanation, discussion, and discrimination. The *STEALTH* results highlight a connection between “better, more honest explanations” and “reducing biased results” into something that we might called “trusted communication”. Are there other such “trusted communication” methods? Would they perform better than *STEALTH*? We do not know– but that would be a useful direction for future research.

ACKNOWLEDGEMENTS

This work was partially funded with support from IBM and the National Science Foundation Grant #1908762. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of IBM or the NSF.

REFERENCES

[1] A. Marketplace, “Machine learning & artificial intelligence,” 2019.
[2] “Wolfram neural net repository.” [Online]. Available: https://resources.wolframcloud.com/NeuralNetRepository
[3] M. Xiu, Z. Ming, Jiang, and B. Adams, “An exploratory study on machine learning model stores,” *IEEE Software*, 2021.
[4] M. Ribeiro, K. Grolinger, and M. A. Capretz, “Mlaas: Machine learning as a service,” in *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2015, pp. 896–902.
[5] S. U. Noble, *Algorithms of Oppression*. NYU Press, 2018. [Online]. Available: http://www.jstor.org/stable/j.ctt1pwt9w5.1
[6] J. Chakraborty, S. Majumder, and T. Menzies, “Bias in machine learning software: why? how? what to do?” in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 429–440.
[7] M. T. Ribeiro, S. Singh, and C. Guestrin, “"why should i trust you?": Explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1135–1144. [Online]. Available: https://doi.org/10.1145/2939672.2939778

[8] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods," in *3rd AAAI/ACM Conference on AI, Ethics, and Society*, 2020.

[9] Z. Chen, J. Zhang, F. Sarro, and M. Harman, "Maat: A novel ensemble approach to addressing fairness and performance bugs for machine learning software," in *The ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2022.

[10] J. Chen, V. Nair, R. Krishna, and T. Menzies, ""sampling" as a baseline optimizer for search-based software engineering," *IEEE Transactions on Software Engineering*, vol. 45, no. 6, pp. 597–614, 2018.

[11] K. Peng, J. Chakraborty, and T. Menzies, "xfair: Better fairness via model-based rebalancing of protected attributes," *IEEE Trans SE, accepted, to appear*, 2022. [Online]. Available: https://arxiv.org/abs/2110.01109

[12] M. R. Hess and J. D. Kromrey, "Robust confidence intervals for effect sizes: A comparative study of cohen'sd and cliff's delta under non-normality and heterogeneous variances," in *annual meeting of the American Educational Research Association*, vol. 1. Citeseer, 2004.

[13] N. Mittas and L. Angelis, "Ranking and clustering software cost estimation models through a multiple comparisons algorithm," *IEEE Transactions on software engineering*, vol. 39, no. 4, pp. 537–551, 2012.

[14] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning*. The MIT Press, 2006. [Online]. Available: http://dblp.uni-trier.de/db/books/collections/CSZ2006.html