FAI: *xPLAIN*– Mitigating threats to fairness posed by semi-supervised multi-goal explanations

Tim Menzies, IEEE Fellow, NC State

It is the ethical duty of software researchers and engineers to produce quality software that makes fair decisions. This is especially true for high-stake software that makes important decisions about human lives. Sadly, there are too many examples of machine learning software exhibiting unfair/biased behavior based on some privileged attributes like sex, race, age, marital status.

In response to these issues, many research teams have developed explanation tools that can reveal the details about the internals of seemingly opaque AI models. Alarmingly, much of that successful work is now threatened by a new generation of algorithms that can obscure discriminatory features of models.

To address that threat, we propose xPLAIN which offers extensions to current methods for (a) fairness testing, and (b) multi-goal reasoning and (c) semi-supervised explanations. Specifically, xPLAIN is a non-parametric Monte Carlo method based on recursive random projections. Given some model that is being assessed for unfairness, xPLAIN3 can quickly infer an approximation to that model, which can be used for the purposes of unfairness testing and mitigation.

This work will deliver new unfairness detection and mitigation methods that will work faster than prior work, as well as defending users against malicious attempts to hide discriminatory practices.

KEYWORDS:

AI; fairness, explanation, semi-supervised learning, multi-goal reasoning.

INTELLECTUAL MERIT:

Much prior work on the unfairness detection and mitigation has made limiting assumptions about the nature of their work. For example, work on software fairness testing assumed extensive access to the model (and the data used to create it). In many scenarios, it may not be possible to grant such extensive access. Hence, we offer tools that support a broader class of problems; specifically: semi-supervised multi-goal learners that work, even if they can access only a small fraction of the data.

BROADER IMPACTS:

We focus on issues of tremendous social importance—the detection and removal of unfair conclusions resulting from the application of AI tools. This work will increase America's ability for industrial and academic innovators to produce more AI-based software as well as certifying that those tools are fair to their users.

As to other impacts, PI Menzies is a member of his department's Broadening Participation Committee (BPC) that actively seeks to (a) *understand* what factors make computer science more (or less) attractive to underrepresented groups; (b) *educate* faculty, staff and students on how different behaviors and activities affect diversity, quality and inclusiveness; (c) *increase* the percent of students who are identified as women; and (d) *evaluate* the success of his department's BPC team in broadening participating. A (small) portion of this grant would be allocated to support these BPC work, focusing especially on the students involved in this grant.

As to other broader impacts, this work will inform the curriculum and lecture notes of the various NC State NSF-funded REUs (research experience for undergraduates) as well as graduate SE classes (taught by PI Menzies). Also, funds from this work will be used to support students attending the Grace Hopper conference, and the Richard Tapia Celebration of Diversity in Computing.

FAI: *xPLAIN*– Mitigating threats to fairness posed by semi-supervised multi-goal explanations

Tim Menzies, IEEE Fellow, NC State

1 Transformative Research

PROBLEM: AI has become dangerously unfair. There are too many examples of AI software exhibiting unfair/biased behavior based on privileged attributes like sex, race, age, marital status. For example, Amazon had to scrap an automated recruiting tool as it was found to be biased against women [16]. Also, a recidivism assessment models used by the criminal justice system was found to be more likely to falsely label black defendants as future criminals (twice as often as white defendants) [11].

AI explanation tools are now a contributing factor to that danger. When explanation tools can succinctly describe some complex model, then:

- It is possible for malicious reasons to hide certain discriminatory details about that complex model.
- It is no longer possible to apply *software fairness testing* (described below). This is troubling since we had thought that significant progress was being made in that area. E.g. as discussed in §2.3. there now exist methods that can mitigate unfairness without damaging predictive performance [32], but only when the testing tools can access extensive information about the model.

HYPOTHESIS: We can transform dangerously unfair explanation tools into algorithms that (a) explain complex models, (b) measure unfairness; and (c) mitigate unfairness. Further, we can do so even when we have very limited interactions with the model being explained.

METHODS: Our xPLAIN semi-supervised learner recursively bi-clusters data via random projections (see Figure 2b). This learner needs very little data to operate. As shown by our preliminary experiments in §2.6, xPLAIN needs only just a few dozen examples to compete with state-of-the-art optimizers (that require thousands of examples, or more). Given that tree, then if existing data is not enough, we can generate new data via non-parametric means (i.e. extrapolating between examples in leaves of the recursively clustered data). Unfortunately, such stochastically generated data is distinguishable from ground truth data. This means that malicious humans could use those differences to recognize when they should lie about the discriminatory properties of their model. To thwart such maliciousness, we will improve the fidelity of our data generation via additional constraints learned from positive learners (see §4.3.3).

EVALUATION: We will test on the fairness case study data sets of Table 1, but in a multi-goal context; e.g. we will seek to maximize recall while at the same time minimizing false alarm as well and all our measures of unfairness (defined in Table 2). For details on that evaluation, see §4.2 (*Research Questions*).

2 Background

2.1 Embedding Innovations in Real Systems

Machine learning methods can be packaged and executed in cloud "containers" (e.g. using tools like opendatahub.io). Many such containers are available for rent, in "model stores" at the AWS marketplace [1] and the Wolfram neural net repository [2]. Users rent these models plus the -CPU needed to run them. They upload their data and, later, the model returns labels (conclusions).

As seen in Table 3, the Xiu et al. survey of Jan'21 found over 300 such models-for-rent [95]. Given the current interest in commercial AI, it is expected that these numbers will soon grow much larger.

Table 3: Number of models in model stores. From [95].

Type	AWS	ModelDepot	Wolfram
Image	61	24	59
Video	13	2	0
NLP	35	5	18
Audio	12	1	2
Structured	107	0	0
Total	228	32	79

These models make decisions that could inappropriately affect different social groupings defined by age, gender, or race. For example, in the AWS model store, *Credit Default Predictor* uses attributes like gender, education, age, and previous history of payments. Assessing the fairness of *Credit Default Predictor* is an important task since such predictors have the potential to perpetuate social inequities (e.g. it is hard to break the cycle of poverty if your zip code prevents you from access a line of credit).

Also in the AWS model store, the *Hospital Readmission* model predicts the probability that a patient will not be readmitted after discharge. This model's inputs include financial class, sex and age. It is important to assess the fairness of *Hospital Readmission* since it is highly inappropriate (and dangerous) to affect a patient's life expectancy due to their (e.g.) social status.

Table 1: **Data sets used in many papers on fairness. As the field of fairness testing evolves, we expect this table to grow much larger. Hence, as part of this work, we would monitor the data sets used in this research arena (and we would test all our methods on all that growing space of data).**

Dataset	#Rows	#Features	Protected Attribute	Description
Adult Census Income [3]	48,842	14	Sex, Race	Individual information from 1994 U.S. census. Goal is predicting income >$50,000.
Compas [10]	7,214	28	Sex,Race	Contains criminal history of defendants. Goal is predicting re-offending in future
German Credit [4]	1,000	20	Sex	Personal information about individuals & predicts good or bad credit.
Default Credit [12]	30,000	23	Sex	Customer information for people from Taiwan. Goal is predicting default payment.
Heart Health [5]	297	14	Age	Patient information from Cleveland DB. Goal is predicting heart disease.
Bank Marketing [13]	45,211	16	Age	Contains marketing data of a Portuguese bank. Goal is predicting term deposit.
Home Credit [14]	37,511	240	Sex	Loan applications for individuals. Goal is predicting application accept/reject.
Student Performance [8]	1,044	33	Sex	Student achievement of two Portuguese schools. Target is final year grade.
MEPS-15,16 [9]	35,428	1,831	Race	Surveys of families, individuals, medical providers, employers. Target is "Utilization".

Table 2: **Definition of the performance and fairness metrics used in this study.**

Performance Metric	**Ideal Value**	**Fairness Metric**	**Ideal Value**
Recall = TP/P = TP/(TP+FN)	1	**Average Odds Difference (AOD)**: Average of difference in False Positive Rates(FPR) and True Positive Rates (TPR) for unprivileged and privileged groups [27]. TPR = TP/(TP + FN), FPR = FP/(FP + TN), $AOD = [(FPR_U - FPR_P) + (TPR_U - TPR_P)] * 0.5$	0
False alarm = FP/N = FP/(FP+TN)	0	**Equal Opportunity Difference (EOD)**: Difference of True Positive Rates (TPR) for unprivileged/ privileged groups [27]. $EOD = TPR_U - TPR_P$	0
Accuracy = $\frac{(TP+TN)}{(TP+FP+TN+FN)}$	1	**Statistical Parity Difference (SPD)**: Difference between probability of unprivileged group (privileged attribute PA = 0) gets favorable prediction ($\hat{Y}$ = 1) & probability of privileged group (privileged attribute PA = 1) gets favorable prediction ($\hat{Y}$ = 1) [30]. $SPD = P[\hat{Y} = 1 \| PA = 0] - P[\hat{Y} = 1 \| PA = 1]$	0
Precision = TP/(TP+FP)	1	**Disparate Impact (DI)**: Like SPD, but it me assures the ratios (not the difference) in probabilities [42]. $DI = P[\hat{Y} = 1 \| PA = 0]/P[\hat{Y} = 1 \| PA = 1]$	1
F1 = 2 $* Prec * Recall/(Prec + Recall)$	1		

So how to trust that model stores are not dispensing discriminatory models that are unfair to certain social groups? This is an important question since:

- Given the ubiquity of the internet, model stores can unleash discriminatory models to a wide audience.
- Currently, the fairness of the models is unknowable. Model store models usually have no tools for (a) measuring or (b) mitigating discriminatory bias.

This second point is particularly troubling since Xiu et al. warn that over 70% of these models are early lifecycle research prototypes. As such, we should as the very least routinely apply some kind of fairness verification process to the models in model stores.

But there is a problem with that. As discussed below in §2.2 and §2.3, current methods for assessing the fairness of a model assume that verification tools can access extensive details about the model. Yet there are valid privacy, pragmatic and proprietary reasons why this may not be possible:

- E.g. model owners want to hide details, least their competitors reverse engineer their model;
- E.g. the data used to build a model may reveal sensitive information about individuals.
- E.g. owners of the models (inside the model store) want to keep those models details in-house, at least until they recoup their development cost via renting out their model.

Even if the models and data were open-source, then supervised methods may still not be appropriate:

- E.g. domain expertise is in short supply so we can only check correctness of a handful of model labels;
- E.g. it might cost too much to pay for (e.g.) running an optimizer, 20 times in order to explore thousands of possible parameter settings for a deep learner.

Hence our goal is to **measure and mitigate model store unfairness**, even if when we have **limited interaction with a model**. We will say we can *query* a *MODEL* to compute y from x:

$$y_1,\ y_2,\ ..,\ y_n = MODEL(x_1,\ x_2,\ ..,\ x_m)$$

For single goal reasoning, $n = 1$. For multi-goal reasoning $n \geq 2$ since models may return multiple predictions about individuals (.e.g, predicted miles per hour and acceleration of a car). For the privacy, pragmatic, and proprietary reasons stated above, we assume we cam access limited details about *MODEL* (e.g. even if we knew that the neural net is CNN or RNN, we probably will not know the hyperparameter settings). Hence we need a model-agnostic method that only reasons about the x, y values.

We say that model store fairness verification studies should be conducted *before* a model is deployed for day-to-day usage. Another way to say that is that we seek to support "try before you buy"; i.e. the process of surveying multiple models to see which best suits the local needs. Given a limited number of queries Q, we may only know the importance of some of the attributes x' (and $|x'| \ll |x|$). In our experience, the APIs to models usually requires all inputs to be specified. Hence, some *complete* function is required to fill in all the x values not seen in x':

$$y_1,\ y_2,\ ..,\ y_n = MODEL(\ complete(x'_1,\ x'_2,\ ..,\ x'_j)\ )\ where\ j < m \tag{1}$$

This *complete* function seems like a minor detail. However, as discussed in §3, this *complete* function will raise significant issues for model fairness (a point we will discuss later, at great length).

2.1.1 Pragmatics

Before going on, just by way of expectation management, we digress to make the following point.

The experiments of this paper will require millions of calls to models as we compare what happens when we have complete versus partial knowledge of that model. For that testing, it would be difficult to directly use real model store models since, as mentioned above, those models may not offer all the data required for this analysis.

We considered loading up to AWS our own models, with built-in discriminatory properties but that is ethically unacceptable (lest someone unwittingly used our discriminatory model in their domain).

Instead, we will build a fake model store inside our own firewalls. We will build and host containers from all the models of Table 1. These containers will be accessed via private APIs inside our own firewalls that track the number of queries made to these models.

2.2 Supervised Methods for Software Fairness Testing

This section describes standard methods for software fairness testing. Note that these methods required extensive interaction with the models under study Later in this proposal, we will explore methods to achieve the same results with far fewer queries.

Proponents of software fairness testing, such as Brun [29], argue that models learned by machine algorithms need to be assessed via *both* performance *and* fairness metrics:

- The left-hand-side of Table 2 shows the standard performance metrics.
- The right-hand-side of Table 2 shows some fairness metrics. These are defined in terms of "protected" attributes and "unprivileged" groups. In the case of credit card application datasets, the attribute "sex" is protected since it contains a traditionally under-privileged group "sex=female". Other protected attributes might include the protected attributes, listed in Table 1 and/or sex, race, age, income, veteran or marital status, other social groupings.

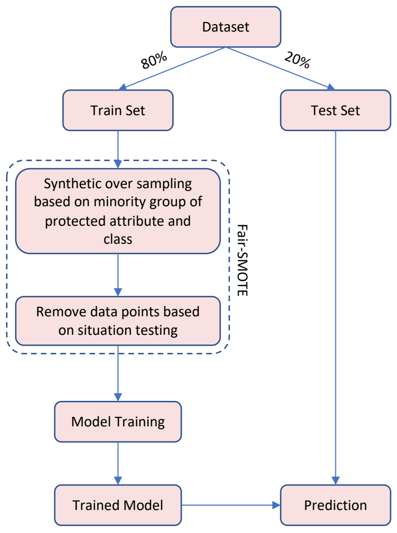
Software fairness testing checks if the conclusions made by a model about unprivileged groups are markedly different from the rest of the population. And after unfairness detection, comes unfairness mitigation. The influence of the privileged attributes on the dependent variable can be reduced in many ways including (e.g.) hyperparameter optimization. For example, PI Menzies, Chakraborty and Johnson et al. [33,51] searched for learner settings to that resulted in improvements to the fairness metrics of Table 2 (Johnson et al. used a very slow grid search while Chakraborty et al. used a faster tabu search).

In some cases, the effects of privileged attributes are “essential”; i.e. “sex” can have an influence on an obstetric diagnosis (e.g. instances with“sex=male” cannot get pregnant). Hence, a repeated observation from the fairness testing literature is that improvements to the fariness metrics also means a deterioration in performance metrics (e.g. see experiments with instance reweighing [52], prejudice remover regularization [54], adversarial debiasing [100], equality of opportunity [45], and a reject option classifier [53] and other techniques [31,45,52–54,100]). Consequently, a pessimistic truism in the software fairness testing community is that, as said by Berk et al. [28]: *“It is impossible to achieve fairness and high performance simultaneously (except in trivial cases).”*

2.3 Recent Success with Fairness Analysis

In 2021, PI Menzies and Chakraborty and Majumder and [32] found they could reverse the pessimism of Berk et al. using their Fair-SMOTE tool[1]. It turns out, that the “Berk effect” (were improvements in either performance or fairness mean degrading the other) can be avoided, if researchers take greater care with how they *remove* and *add* instances to the training data.

Figure 1: Fair-SMOTE. From [32]. Note that Fair-SMOTE takes care to isolate the test from any transformations applied to the training set.


Dataset
80%
20%
Train Set
Test Set
Synthetic over sampling based on minority group of protected attribute and class
Fair-SMOTE
Remove data points based on situation testing
Model Training
Trained Model
Prediction

As to *removing examples*, Fair-SMOTE carefully prunes discriminatory training data, as follows. Situation testing [6,7] is a legal tactic that defines “discrimination” as similar individuals getting different outcomes. Using a logistic regression model, Fair-SMOTE trains a preliminary model to makes predictions for all instances. The privileged attribute values for every row of training data are then flipped (e.g. male to female, white to non-white). If that change alters predictions, then that row is removed. For the data sets of Table 1, this removed 8% of the examples (median value). Note that in Fair-SMOTE, situation testing follows example creation (described below), so any discriminatory examples generated via example mutation are removed.

As to *adding examples*, Fair-SMOTE takes care to rebalance not just class frequencies (as done with SMOTE [35]) but all ranges of protected attributes. For example, suppose some original training had 20% of data with a criminal record (holding $\frac{4}{5}$ths male and $\frac{1}{5}$ths female) and 80% non-criminal (holding $\frac{6}{20}$ths male and $\frac{4}{10}$ths female). Traditional rebalancing methods (like SMOTE [35]) would generate data with 50% criminal (holding $\frac{4}{5},\frac{1}{5}$th men,women) and 50% non-criminal (holding (holding $\frac{6}{10},\frac{4}{10}$th men,women). Fair-SMOTE’s rebalancing, on the other hand, would generate data with 50% criminal (half of which would be men and half of which would be women) and 50% non-criminal (half of which would be men and half of which would be women)[2].

Table 4 compares Fair-SMOTE to traditional SMOTE [35] as well as a prior state-of-the-art algorithm. Optimized Pre-processing (OP) by Calmon et al. [31] tries to find, then fix, unfairness. OP is a data-driven optimization framework for probabilistically transforming data in order to reduce algorithmic discrimination. OP treats bias mitigation as a convex optimization problem with goals to preserve performance and achieve fairness. We chose this work as a baseline because, like Fair-SMOTE, it is also a data pre-processing strategy. Note that contrary to the pessimism of Berk et al.:

- Compared to SMOTE, Fair-SMOTE achieves similar performance and better fairness scores.
- Compared to OP, Fair-SMOTE achieves better performance scores and similar fairness scores.
- More importantly, there is no evidence in Table 4 of the “Berk effect” where improvements in either of performance or fairness mean degrading the other.

[1] That paper won a distinguished paper award at the 2021 ACM SIGSOFT Joint European SE Conference on the Foundations of SE.

[2] In our experience, changing class frequencies in this way can confuse some algorithms such as the Naive Bayes class likelihood calculation of $P(H)\times\prod_i P(E_i|H)$. This is particularly true especially in the case of only a few attributes (i.e. i is small). But once the feature size grows to more than half a dozen attributes, then the contribution of $P(H)$ is less important than the $\prod_i P(E_i|H)$ term (which means class rebalancing no longer confuses Naive Bayes).

	Recall	False alarm	Precision	Accuracy	F1 Score	AOD	EOD	SPD	DI	Total
	SMOTE [35] vs Fair-SMOTE [32]									
Win (for Fair-SMOTE)	4	4	1	6	3	33	33	34	32	150
Tie	25	27	29	28	30	2	3	2	2	148
Loss	7	5	6	2	3	1	0	0	2	26
Win + Tie	29	31	30	34	33	35	36	36	34	298/324
	Optimized Pre-processing [31] vs Fair-SMOTE [32]									
Win (for Fair-SMOTE)	10	7	4	3	12	1	2	2	3	44
Tie	21	22	26	30	20	34	33	32	31	249
Loss	5	7	6	3	4	1	1	2	2	31
Win + Tie	31	29	30	33	32	35	35	34	34	293/324

Table 4: **Fair-SMOTE performs as well, or better, than alternative methods. Results from 50 repeats of Figure 1 on the data of Table 1 where the learner was logistic regression. Pink cells denote results of non-parametric significance and effect size tests (bootstrap and CliffsDelta) where Fair-SMOTE perform statistically better than anything else. From [32].**

2.4 Analysis: Can we recommend Fair-SMOTE?

The above results seem to show good progress on the problem of measuring and mitigating unfairness in AI. Alarmingly, much of that success is now threatened by a new generation of semi-supervised explanation algorithms. Such algorithms explore models of the form $y = MODEL(x)$. That exploration uses public domain knowledge of x plus a very small number of queries to the model (to generate some sample y values). As we shall see, when we only run a few queries on the model, then this can lead to (1) **unfairness measurement problems**, (2) **unfairness mitigation problems**, and (3) **problems with malicious activity** that tries to hide discriminatory models. In §2.5, we offer some details on semi-supervised explanation algorithms. After that, §3 offers more details on these three problems and §4 sets out a research agenda to address these problems.

Before beginning we offer two notes:

- In our own prior work, we never realized the connection between the fairness testing and mitigation (discussed above) and semi-supervised explanation (discussed below). But recently it has become clear that "under-the-hood", both these methods do the same thing; i.e. *both methods explore the structure of the data*. Hence, as we shall see, current methods for semi-supervised explanation can be extended to include unfairness measurement and mitigation.
- When discussing this work with colleagues, they sometimes comment that our software is more a "planner" (on what to change) than an "explaination" device. In reply, we say much precedent in the AI literature for connecting planning to explanation. Adadi and Berrada [20] list four main motivations for building explanation systems: (a) *explaining to justify* decisions made by some model; (b) *explaining to control* a system, allowing its debugging and the identification of potential flaws; (c) *explain to improve* predictive performance and/or efficiency; and (d) *explaining to discover* novel relationships and patterns in the data. As shown by the examples below, our "planning as explanation" methods address motivation (b,c,d). Also, in their systematic literature review on AI and explanation, Violane et al. [92] report is that one of the most widespread use of explanations in AI is to find ways to change a models behaviour; e.g. to debug a system in order to stop some bug reoccurring. We find these usages of "explanation" to be consistent with our "explanation-as-panning" algorithm.

2.5 Fairness and Explanation: Why? How?

The first obvious question is this: why discusses explanation and AI fairness?

Many professional, industrial, and governmental organizations use concepts related to explanation when they discuss ethics, AI and fairness. For example, three recent white papers on AI and ethics from the IEEE [18], Microsoft [19] and the European Union [17] all reference accountability and transparency. That is, all these organizations are saying that fair and ethical AI must support *explanations* that allow users to understand what is going on inside the AI. According to Rudin [84], explainability is a core concept for model fairness: *"Many of the ML models are black boxes that do not explain their predictions in a way that humans can understand. The lack of transparency and accountability of predictive models can have (and has already had) severe consequences (in a large number of domains)".*

Rudin [84], also argues that, when dealing with explanation, that *simpler* is *better*. Minimal explanations are not just convenient, they may also be essential. Cognitive psychologists such as Larkin et al. [64]

characterize human expertise in terms of very small short term memory (used as a scratchpad for current observation) and a much larger long term memory (that holds the rules that process the short-term memory). Short term memory is very small, perhaps even as small as four to seven items [36, 78][3]. Experts are experts, says Larkin et al. [64] because the patterns in their long term memory dictate what to do, without needing to pause for reflection. Novices perform worse than experts, says Larkin et al., when they clog their short-term memory with too many observations. Hence, according to Larkin et al., short explanations that do not clog short term memory, and tell directly what to do without further reflection, are a necessary pre-condition for achieving expert-level performance.

In our experience, minimality is a necessary, but not sufficient, condition for good explanations, especially when exploring multiple goals. For example, suppose we need to explain what kind of cars satisfy three goals: fuel-efficiency, good acceleration, and lightweight (in order to minimize the materials required to manufacture it). The file *auto93* (from the UC Irvine ML repository) has hundreds of examples of cars that mention those features as well as the number of cylinders c, horsepower h, displacement d (a.k.a. size of car), year of release y and country of origin o. Using that data, we can learn Equation 2 which could be used to explain how to improve some *disjunction* of weight *or* mph *or* acceleration.

$$\begin{aligned} weight &= 60.7\times c + 5.2\times d + 4.1\times h + 13.7\times y + -48\times o + 234.2 \\ mpg &= -0.6\times c + -0.1\times d + -0.1h + 0.6\times y + 1.2\times o + -12.9 \\ accel &= 0.1\times c + 0.1\times d + -0.1h + 0.1\times y + -0.2\times o + 21.2 \end{aligned} \tag{2}$$

That said, the linear model of Equation 2 may not satisfy Larkin et al.'s requirement that expert-level explanations must tell us directly what to do without further reflection. Suppose we want to use Equation 2 to generate an explanation about what to do in order to improve the *conjunction* of weight *and* mph *and* acceleration. Unlike before, this is now a *mulit-goal* problem that must trade-off any competing effects between our three goals. This can be a surprisingly complex and slow process. For example, NSGA-II [38] is a standard tool for multi-goal optimization that could perform that trade-off. To do so, NSGA-II would (a) mutate population of cars, (b) scores the mutants using the three equations of Equation 2, then (c) select the best scoring mutants as parents for the next generation (denoted $G_{i+1} \in G$) of mutants. Starting with equations not much more complex than Equation 2, PI Menzies and his graduate student Kewen Peng [56] applied NSGA-II to learn multi-goal explanations for $C = 10^4$ software configuration options. In that application, the $O(GC^3)$ non-dominating sort procedure of the NSGA-II [38] took hours to terminate.

2.6 Semi-supervised Explanations with xPLAIN (and xPLAIN2)

While NSGA-II has issues with generating explanations from models like Equation 2, the same is not true for other methods. In this section, we explore those other methods, under the following assumptions. As clients of the model store, we cannot directly access the internal data (for the privacy or pragmatic or proprietary reasons stated in §2.1). That said, we can assume access to the publicly available information about the number and type and range for each x value (since if were otherwise, then we could not even call the model). We can also assume some knowledge of the constraints on x. For example, we might know what x_i values imply or exclude other x_j values (e.g. if *sex=male* then *obstetric history* must be nil). In defense of this assumption:

- Data scientists from a specific domains, often learn patterns of expected (and unexpected) x values.
- Those data scientists might have also have representative samples of the x values.

Using that publicly available knowledge, semi-supervised learners can explore a space of possible x values, while occasionally make a few called to *MODEL* in order to obtain some y values. *Semi-supervised learners* [34] train their models by combining a small amount of labeled data with a large amount of unlabeled data during training. *Semi-supervised explanation algorithms* use semi-supervised learning to generate very small explanations [50, 69, 101]. For example, in the case of Equation 2, such explanations would plan how to select cars with the most fuel efficiency, least weight, and most acceleration.

Semi-supervised algorithms [34] utilize *manifold*, *continuity*, and *clustering* assumptions. The *manifold* assumption is that data lies approximately on a manifold of a much lower dimension than the input space.

[3] Recently, Ma et al. [70] used evidence from neuroscience and functional MRIs to argue that STM capacity might be better measured using other factors than "number of items". But even they conceded that "the concept of a limited (STM) has considerable explanatory power for behavioral data".

Figure 2: Recursive random projections via FASTMAP [41].

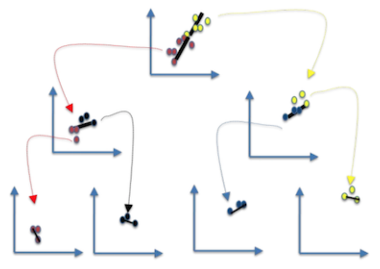


Figure 2a.

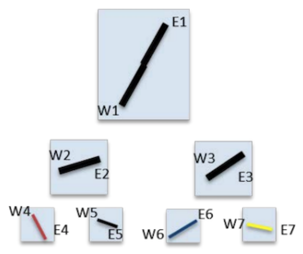


Figure 2b.

NOTES: If M is any example (selected at random) then E (east) is the example most distant from M and W (west) is the example most distant from E. Let $c = dist(E, W)$ and let all other examples have distances a, b to E, W, respectively. Using the cosine rule, we say $x = (a^2 + c^2 - b^2)/(2c)$, is the distance x where an example falls on a line drawn from E to W. By splitting data on median x, the examples can be then bi-clustered. xPLAIN algorithm can then (e.g.) recurs on both clusters to find an approximation to the data manifold, shown in Figure 2b.

Under this manifold assumption, higher-dimensional data is approximated in a much lower dimension space, with little or no performance loss [66]. When data is spread over just a few underlying dimensions, then there are fewer ways that examples can differ. Hence, there is more *continuity* between nearby examples and we do not need to reason separately about each example. Rather, we can *cluster* similar examples and reason about one item per cluster.

Using those assumptions, there are many ways to extrapolate from a small number of labels to a larger set of data [104]. For example, the *xPLAIN algorithm* [69] (discussed in the this section) recursively bi-clusters the data down to $\sqrt{N}$of the data, asking only for the labels of the two most distant examples at each label of the tree (and the final labels are the leaf clusters at the bottom of that cluster tree) Also, *self training* algorithms [97] incrementally guesses new labels from a learner trained on all labels seen to date. Further, *GMM with expectation-maximization* algorithms [47] use a Gaussian mixture model to clusters the data (and use those clusters to label the data). Furthermore, *label propagation algorithms* [103] guess labels using a majority vote across the labels seen in nearby examples (or clusters). Label propagation algorithms never update their old labels. *Label spreading* algorithms [102], on the other hand, update old labels using with feedback from subsequent labelling. Label spreading algorithm iterates on a similarity matrix between example and normalizes the edge weights by computing the normalized graph of the Laplacian.

Table 5: Notes on domination predicates for multi-goal reasoning.

When dealing with 1 goal, a simple ≤ function can rank goal values of E (east) and W (west). But for multiple-goal reasoning, examples must be ranked across many goals values.
Binary domination says E is better than W if E has at least one better goal value (and zero worse goal values) than W.
For more three or more goals, binary domination has trouble distinguishing examples [93], in which case Zitler's *continuous domination* predicate [105] is recommended [86].
Continuous domination extends binary domination by summing the actual difference in goal scores.
For individuals x, y have n goals variables (each of which has been normalized 0..1, min..max, to x', y'). Zilter says X is better than Y if the mean loss moving X to Y is less than the mean loss moving Y to X; i.e. $better(x, y) = Loss(x', y') < Loss(y', x')$ where $x' = norm(x)$, and $y' = norm(y)$ and $Loss(a, b) = -\sum_i^n k^{w_i*(a'_i-b'_i)/n}$ where k is some constant (usually 2.7183) and $w_i = -1$ if we want to minimize goal i (otherwise, $w_i = 1$).

Of all the methods in the last paragraph, we use xPLAIN since as shown by the experiments described later in this section, xPLAIN is very effective at exploring large data sets with just a few queries. Also, the recursive bi-clustering used by that approach can also be used as a non-parametric sampling method (by extrapolating between examples in leaves of the recursively clustered data). Later in this proposal, we will make extensive use of that process.

xPLAIN recursively bi-clusters the data using the FASTMAP [41] random projection algorithm. As shown in Figure 2, xPLAIN finds two distance points *east,west* (denoted E, W). The rest of the data is then divided into two, depending on whether or not each example is closer to E or W. The algorithm can then recurse on either half. For example, we can prune the worst half of the data then recurse on the rest.

xPLAIN produces short explanations since it does not try to explain all the data. Rather it only explains differences between east-west pairs in non-leaf nodes. For the tree of Figure 2b tree, there are only three such differences. For a tree of $N = 10,000$ examples with leaves of size $\sqrt{N} = 100$, there are only 63 such differences. Further, a greedy search that prunes half the data at each step of the $N = 10,000$ tree, will arrive at the leaves after explaining just six differences.

xPLAIN is a multi-goal reasoner, that determines which of E, W is "worst" via a *domination predicate* (see Table 5). As it recurses, xPLAIN uses *minimal contrast sets* to explain why it is selecting data. Minimal contrast sets are the smallest contrast sets that most distinguish two sets of examples. After dividing the data in two (using FASTMAP), xPLAIN finds the minimal contrast sets that (a) most distinguishes the two halves; then (b) prunes the worst half of the data. To do so, xPLAIN discretizes numeric attributes via the Chi-merge algorithm [55]. Next, it uses a Naive Bayes classifier trained on the two halves of the data to rank ranges by their likelihood of being labeled as the better half. The top-ranked range is then printed to screen. xPLAIN then recurses on the better half of the data (stopping at $\sqrt{N}$ of the original data).

Figure 3 shows data selected by xPLAIN for *auto93*. Here, xPLAIN recursed twice (so $Q = 2$) and reported two minimal contrast sets: "cylinders $\leq$ 4" and "origin==3". Note that xPLAIN's explanations satisfy Larkin et al.'s requirement (from §2.5) that explanations tell us directly what to do without further reflection.

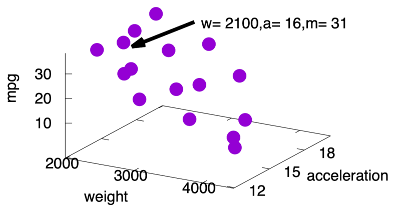


Figure 3: 16 clusters of 398 examples from *auto93*. Arrow shows data found by XPLAIN.

Further improvements are possible; e.g. in Figure 3, the acceleration is still only mid-range in the selected data. PI Menzies and Lustosa [69] have explored a variant of xPLAIN (called xPLAIN2) that builds the tree of Figure 2.a before pruning. Contrast sets are computed for every non-leaf node (using minimal contrast sets) for every non-leaf node. The contrast set that best separates the data at a particular node is then selected as the *pruning constraint*. This is used to prune one of the subtrees under this node plus all examples in the rest of the tree that contradicts the settings in the pruning constraint. xPLAIN2 then loop, using the reduced data set (stopping at $\sqrt{N}$ of the data).

PI Menzies and Lustosa [69] have compared xPLAIN2 to state-of-the-art search-based software engineering methods [25] for exploring 10,000 examples from (a) project management data with 128 binary features; (b) an online e-billing system; and (c) numerous artificial data sets where the number of features (and constraints between those features) can be determined by the experimenter. They found that xPLAIN2 could explore tens of thousands of examples in models with 100+ attributes via just a queries ($Q \leq 20$) data labels. Further, when compared to state-of-the-art search-based SE (SBSE) methods [25], xPLAIN2 found best results (i.e. examples were the labels reflect more desirable domain results).

Note that the success of xPLAIN2 can be explained via the assumptions of semi-supervised learning; i.e. to reason about data, just reason about the underlying manifold (e.g. the Figure 2b structure).

2.7 Analysis: can we Recommend xPLAIN2?

The above results seem to endorse the use of semi-supervised explanation algorithms:

- xPLAIN2 is useful for reasoning about models for which you have limited information (this is useful if the *MODEL* is in a model store, where most of it its details are buried away behind an API).
- Using xPLAIN2, it is possible to build an approximation of *MODEL* without having to access all the details of the data used to train *MODEL*. This can be done with very few queries to model *MODEL* which means that it would be relatively inexpensive to use xPLAIN2 to certify that model *MODEL* is useful (prior to spending considerable time and resources adding in *MODEL* to some local data-mining pipeline). Hence, xPLAIN2 would be a useful tool to "try before you buy"; i.e. for surveying multiple models to see which best suits the local needs.
- xPLAIN2 works for a single goal and multiple-goal reasoning. It can be used for classification problems as well as intricate optimization problems that must trade-off between competing factors.
- xPLAIN2 is a model-agnostic tool that could be applied to multiple models (since it operates on the inputs x and outputs y without needing internal model details).

That said, it may be premature to celebrate the success of xPLAIN2. In the next section, the following details about this example will become very important. As we shall see, Detail #1 can have disastrous consequences for fairness (since it lets malicious humans hide the discriminatory properties of their models). But Detail#2 offers a possible way to repair the problems introduced by Detail#1:

- **Detail #1**: The *auto93* data contains five independent x variables but xPLAIN's output just uses two (cylinders and origin). Most models need all the x inputs to be specified before they can compute $y = MODEL(x)$. That is, for xPLAIN (and xPLAIN2) to check the effect of just applying "cylinders ≤ 4" and "origin==3", it needs the *complete* function from Equation 1 to guesses all the x_i values that are present in the model inputs, but missing from the reported contrast sets.
- **Detail #2:** xPLAIN2 returns an approximation to the data manifold. That approximation takes the form of a tree defined recursively by a set of east, west pairs E, W (see Figure 2.b).

3 Research Questions

We say that the following research questions address major problems for any semi-supervised method (not just xPLAIN), especially those that use the *complete* function from Equation1 (see page C3 of this proposal).

3.1 RQ1: How to Address Unfairness Measurement for Semi-Supervised Systems?

Consider some model of the form $y = MODEL(x)$. From the above, we know that xPLAIN and xPLAIN2 make very few queries to this model. That is, anyone running those algorithms will collect very few y values. This is a problem since the unfairness metrics of Table 2 are measurements over a population of x, y values. Hence, the fewer the y values the less accurate are those unfair measures.

3.2 RQ2: How to Address Unfairness Mitigation for Semi-Supervised Systems?

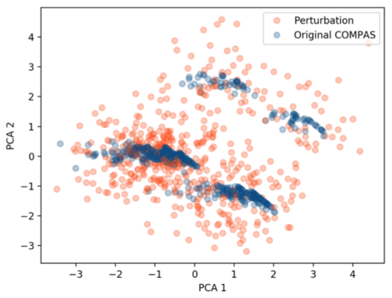
Another problem is that after measuring unfairness comes unfairness mitigation. xPLAIN and xPLAIN2, as defined above, are not mitigation tools:

- The fairness mitigation methods of §2.3 need to rebalance the distributions in a population of x, y values. Here again, since xPLAIN samples very few y values, then there would be little to rebalance.
- More fundamentally, after rebalancing the data, the fairness mitigation methods of §2.3 must remove conclusion bias by rerunning the learners to build another model. But since the premise of this work is that we have no authority over the activity inside the model store, that is clearly impossible.

3.2.1 RQ3: How to Defeat Malicious Model Operators when using Semi-Supervised Systems?

Slack et al. [87] have shown that explanation algorithms are vulnerable to malicious human operators trying to hide discriminatory models. In their work, they discuss explanations generated via stochastic methods. For example, the model-agnostic LIME explanation algorithm [83] generates random instances near to some example. After building some linear models from that generated data, LIME can explain what factors can change the classification of a particular example.

Figure 4: Red/blue = original/invented data. From [87].



The problem here is shown in Figure 4. Here, Slack et al. [87] show the raw data (in red) and data synthesized by LIME (in blue). While the blue dots are near the red dots, Slack et al. show that they are distinguishable via automatic classifiers. That is. the model can distinguish (a) queries-about-explanation from (b) normal-queries seen in day-to-day operation. This means that malicious operators could lie about discriminatory properties of their model just be offer model to queries-about-explanation and another model to the normal-queries seen in day-to-day operation.

When we repeated the analysis of Figure 4, we found that xPLAIN2 has the same problems reported by Slack et al. Recall from §2.3 that our semi-supervised reasoners run a limited number of queries. Hence, they may only know the importance of some attributes x' (and $|x'| \ll |x|$). But the the APIs to models usually requires all inputs to be specified. Hence, a *complete* function is required to fill in all the x values not seen in x'. The output of xPLAIN2's complete function looks like Figure 4; i.e. it is possible for malicious humans to lie to xPLAIN about the discriminatory properties of their models since they can distinguish queries arising from a *complete* function queries from the normal-queries seen in day-to-day operation.

4 Research Plan

4.1 Schedule

Recall from the above that we pose three research questions:

- In RQ1, we ask how to measure unfairness using the unfairness metric of Table 2, even when only some of the data is available (i.e. when running semi-supervised learners).
- In RQ2, we ask how to mitigate unfairness. To check the success of that research, we need to be able to measure unfairness. Hence, RQ2 should follow RQ1.
- In RQ3, we ask how to hide of *complete* function output within the rest of the x data. Before attempting that, we should first check if need to change our semi-supervised methods (and, more specifically, the *complete* function. Hence, RQ3 should follow the rest of the work.

Table 6 shows a schedule for this work. In our experience, this kind of research is CPU intensive, especially when validating a large number of candidate methods. Hence we guesstimate a year worth of work for these research questions.

	Year		
	1	2	3
RQ1: How to Address Unfairness Measurement for Semi-Supervised Systems?	x		
RQ2: How to Address Unfairness Mitigation for Semi-Supervised Systems?		x	
RQ3: How to Defeat Malicious Model Operators when using Semi-Supervised Systems?			x
Collaboration work		x	x
BPC work (broadening participation in computing)	x	x	x

Table 6: Schedule for this work.

As to the *BPC work* line item in Table 6, NSF proposal guidelines encourage the use of grant funding to support initiatives aimed at broadening participation in computer science. Accordingly, we will gratefully use a (small) portion of this funding to understand the BPC activities described in §6.3.

As to the *collaboration work* line in Table 6, our *Collaboration Plan* lists colleagues from Microsoft, Facebook and IBM who have expressed interest in this work. It would be premature to state the precise details of that collaboration until we have running prototypes for the RQ1 and RQ2 work (which we estimate will be available sometime in mid 2024). At this time, what we can say is that our colleagues at Microsoft, Facebook, and IBM have expressed interest in this work and would, in 2024, be willing to review our prototypes, then explore options for unpaid collaboration. Apart from (potentially) being able to access models behind the firewalls of model stores, an important facility these collaborators could offer is access to the subject matter experts that can test the veracity of our artificially generated models.

4.2 Evaluation

In all our work, the success criteria here will the same as for software fairness testing. That is, we seek to measure unfairness and then mitigate it *without* running foul of "Berk effect" discussed in §2.3 (that is, an improvement in either performance or fairness does not mean degrading the other). Further, we must so do without needing so many queries and we must *not* give malicious actors the opportunity to lie to us their models.

For full details on the evaluation, see the next three sections.

4.3 Methods (including Details on Evaluation of Success Criteria)

4.3.1 RQ1: How to Address Unfairness Measurement for Semi-Supervised Systems?

Approach:	Treat unfairness measurement for semi-supervised systems as a *convergence problem* for the fairness measures of §2.
Novel Technology	• Non-parametric sampling method using the manifold discovered by semi-supervised learning. • Speed up convergence with sequential model optimization.
Success Criteria	It can be shown that we $Q \ll N$ queries can calculate nearly the same performance and fairness values found using $Q = N$ values (measured using the Table 2 metrics).

Our RQ1 experiments will set a budget of Q for the number of queries $1 \leq Q \leq N$, where N are the number of data points used to build a model (and note that the software fairness testing methods of §2.3 assume $Q = N$). We will check how quickly our guess made at $Q < N$ the fairness metrics converge to the values we can compute at $Q = N$.

In order to speed up that convergence, we will test if sequential model-based optimization (SMBO) can select useful next queries which speed up convergence. Note that PI Menzies has much recent success with SMBO techniques [79].

Sequential Model-based Optimization (SMBO) is an active learner that reflects on the evidence seen to date to select the best example to label next [79, 106]. Given a tiny number of initial queries, SMBO learns a small model that can report not just mean predictions, but also a measure of variability around that mean. SMBO then uses this information in a *acquisition function* that tells it where to sample next. E.g. that acquisition function might recommend sampling data from a cluster that exhibits the highest mean *and* highest variance. Using SMBO, we will use the queries Q seen too far to select the most informative next query Q_{i+1}. That selection will need two components:

- A measure of "most interesting". We will say that the "most interesting" next query is the one that could most change the performance and fairness metrics seen so far.
- And a space of possible queries. We will generate that space via non-parametric methods that utilize data extrapolation. Specifically, we will use the data manifolds seen in the leaf nodes of Figure 2b.

As said above, we assume that (a) y values are hard to collect but that (b) there is enough publicly available information to collect a useful set of x values. These x values will be hierarchically clustered (see Figure 2b). Whenever any y values become available (from a call to $y = MODEL(x)$) then the cluster containing the x values will be tagged with one more y value.

When SMBO needs to explore data candidates, we will build that data using the leaves of Figure 2b:

- Given leaves of size n_1, n_2.. etc build from $N = \sum_i n_i$ examples, we select leaves at probability n_i/N.
- Within that leaf, we will repeatedly build a mutant by extrapolating between *east, west*
- Let a, b be two x_i values drawn from *east,west* For continuous attributes this mutant will contain the i-the value $x_i + R(a - b)$ where $0 \le R \le 1$ is a random number.
- For discrete attributes, this mutant will contain either a or b at probability 50%.
- For ordinal attributes, this mutant will find all values in the leaf between (and including) a to b. The mutant will then contain any value in that range, selected at probability equal to the percent frequency of that range in this leaf.
- Once the x_i values of the new mutant is created, we will then find its two nearest neighbors with y values within that leaf. The mutant's y values will then be calculated by extrapolating between these neighbors (weighting that extrapolated value by the distance between the mutant and the other example).

Readers of the optimization literature will recognize this as Storn's differential evolution mutation operator [88]. For continuous variables, if R is only recomputed once per mutant, then this mutator *extrapolates all the variables by the same amount*. That is, if there exists some average case association between the pre-mutated examples, then that association is preserved in the mutant.

Just to say the obvious, early on in semi-supervised learning, the mutants generated from this tree will be only approximately correct (and as sampling continues, that approximation will improve).

4.3.2 RQ2: How to Address Unfairness Mitigation for Semi-Supervised Systems?

Approach:	Replace the original *MODEL* with a fairer surrogate *MODEL′*.
Novel Technology	• Apply Fair-SMOTE to the data generated by RQ1. • Extend the SMBO technology from RQ1 to maximize for unfariness reduction. • (Optional) Learn a classifier that can recognize at-risk inputs. Only call the surrogate model *MODEL′* on the at-risk inputs. • Explore semi-supervised learners other than xPLAIN.
Success Criteria	*MODEL′* performs as well as *MODEL* but with reduced unfariness measures (measured using the Table 2 metrics).

We propose two approaches based on (a) sequential model-based optimization (SMBO); (b) a post-processor addon; and (c) other semi-supervised learners.

RQ2a– using SMBO: In this approach, we propose xPLAIN as a bias filter: if we cannot make patches on the original model due to privacy or authority issues, we propose our work as a wrapper API where users can reverse the unfair decisions of the original model by passing it into our API first. In summary, we plan to use SMBO to build a fairer surrogate model that behaves similarly to the original model but is less biased.

As with RQ1, the approach to RQ2 uses the data manifold structure seen in the leaves of Figure 2b. This method will input publicly available x value information and some y values collected from a model:

- Let the model store model and data be *MODEL* and *DATA0*.
- Let the data generated via the sampling methods developed from RQ1 be *DATA1*
- Let *DATA2* be the *DATA1* data, rebalanced by Fair-SMOTE.
- Let *MODEL′* be the model learned from *DATA2*.

We anticipate that some modifications will be required to the methods of §2.3. Specifically, instead of rebalancing range frequencies, it might do better to balance within the geometric space of the manifold of the leaves of Figure 2b. What we anticipate is that there will be a non-trivial interplay between the SMBO methods of RQ1 and this work in RQ2. That is, we might need to consider this RQ2 task also as a *convergence problem*, and then apply SMBO here. In that approach:

- The goal of RQ2's SMBO is to look for regions of the data where xPLAIN predicts to be most likely to have biased conclusions.
- RQ2's xPLAIN converges when no more new biased areas are found. Then xPLAIN will rebalance all the biased regions discovered so far.

RQ2b– using a post-processor: Note also that there is another possible approach for RQ2: instead of replacing the model store model, we could run out of methods as an optional post-processor. In this approach, we use the data generation methods of RQ1 to build a data set, from which we build a classifier predicting what kinds of inputs are most prone to unfairness. For those inputs and only those inputs, we alert some second oracle of a problem. That oracle could be the model owners or, indeed, another model. perhaps even the model built using the methods of RQ2a. The advantage of the latter approach is that:

- In the usual case, the model in the model store makes all the conclusions;
- And only in the problematic cases do we go to the (possibly approximate) methods of RQ2a.

RQ2c– using other semi-supervised algorithms Yet another approach is to divide xPLAIN into its two main functional components, then explore alternates to those components. xPLAIN can be used for (a) semi-supervised label generation and, as described above, (b) data generation. As discussed in §2.6, there are many other ways to implement semi-supervised learning including *self training* [97], *GMM with expectation-maximization* [47], *label propagation* [103] and *label spreading*. In this part of the work, we would test what happens is xPLAIN's label generation is swapped out with these other methods.

4.3.3 RQ3: How to Defeat Malicious Model Operators when using Semi-Supervised Systems?

Approach:	Augment the *complete* function with better knowledge of domain constraints.
Novel Technology	Apply *positive learning* to find those constraints.
Success Criteria	We cannot find classifiers that can distinguish the output of this new *complete* function and ground truth.

In this part of the research, we must stop malicious humans lying to us about the discriminatory properties of their models.

Recall from the above that when we use the *complete* operator of Equation 1, we generate queries that can be distinguished from the normal queries seen in day-to-day operation. That is, model operators can detect when they are being probed by semi-supervised learning. This allows them an opportunity to lie (which they can do by giving different results to those results generated in normal day-to-day operation).

The task of RQ3 is to change the *complete* function such that its outputs are indistinguishable from normal day-to-day operations. *Complete* function output differs from day-to-day queries when its choices made for x values do not reflect the structure of the data. To repair the *complete* function, such that its output most represent the x space, we would use *positive learning*.

Positive learners are those that *only* learn models from positive examples– in our case, that would be x values with no associated y values. The idea behind positive learning is to replace traditional notions of recall and classification with "average case property discovery" and "anomaly detection". Positive learners build a representation of what is usually present (the average case properties). Given enough examples of x values, such positive learners could then learn the average case constraints associated with those projects. These constraints could then be used as a sanity checker on the output of the *complete* function;

i.e. we would keep re-running *compete* until it generated x values that matched the domain constraints.

In turns out there are any number of positive learners that could be applied to this task. The AI literature offers many algorithms that learn using only positive examples [21,24,26,37,39,40,48,49,57,58, 65,80,82,85]. For example:

- Neural net methods can find associations between unlabelled data using, e.g. the self-organizing maps of the word2vec algorithm.
- One-class SVMs have been applied extensively in AI [57,58]. A standard SVM learns a *hyperplane* that separates examples of different classes. With some modifications to its optimization goals, the same tool can learn a *hyper-sphere* that encloses a large percentage of the positive examples seen to date. Such a one-class SVMs have been applied extensively in AI [57,58] to recognize things that are different from the positive examples where those new things fall outside of the hyper-sphere.
- Yet another kind of positive learning is ModelSeeker [26] that found the constraints underlying the scheduling of the Bundesliga (the German Football Liga) from a single schedule. ModelSeeker uses a stochastic hierarchical cluster approach to find average case properties. When exploring a matrix $n * n$ (showing that setting were found together), the algorithm hunts for subsets $x \subset n, y \subset n$ such that most of the rows or columns or diagonals of the $x * y$ matrix had constant values. In the case of constant rows or columns, that means some spurious attribute can be removed. In the case of a constant diagonal, that means a synonym has been discovered which means that those variables can be replaced by a synthesized variable. After these replacements, the algorithm recurses looking for other average-case properties.

Finding useful positive learning would require some experimentation. To test if we have found the right positive learner, we would adopt the methodology of Slack et al. [87]. That is, we would compare real data and artificially generated data (where the later comes from a *complete* function augmented with the constraints learned from a positive learner). These two data sets would be mapped into some lower-dimensional PCA space. Some classifier would then be applied to try and distinguish these two data sets. RQ3 would be a success when that classifier cannot distinguish between the two data sets. To make that comparison valid, we would spend much effort exploring the classifier used in the above test. Symbolic and neural alternatives would be used (augmented with and without discretization, feature selection (or synthesis), and hyperparameter optimization).

The work described for RQ3 is the most ambitious for this proposal, so it is worthwhile discussing what happens if we fail at this task:

- The premise of the work in RQ3 is that we can derive enough domain constraints from publicly available x information. This, of course, may not work without augmenting that publicly available information with extra information gleaned from (say) subject matter experts. And even then, the effort required to find and interrogate those oracles might take too long.
- Hence, while the attempt is worthwhile (since if it works, it will be safer to use model store), if RQ3 fails, we can retreat to the methods of RQ2; i.e. switch to a surrogate data set (rebalanced for fairness).

5 Related Work

Our work takes inspiration from Ji et al. [50] who explore a problem similar to ours. They focus on "an important yet understudied aspect of the fairness problem: reliably assessing how fair a black box model is, given limited labeled data". To address fairness, they use Bayes sampling on the available data to extrapolate a much bigger data set. Our work extends the Ji et al. methods in two significant ways.

- Firstly, the meta-review of the Ji et al. paper from NIPS'20[4] commented that "(their paper) is conceptually and mathematically sound. The significance of the contribution (an auditor tool only, instead of an auditor plus a mitigation tool) is however at the low side". To say that another way, after unfairness measurement should come unfairness mitigation. Accordingly, we will check if the software unfairness measurement and mitigation methods of §2.3 can be applied to the extrapolated data.
- Secondly, Ji et al.'s methods make numerous parametric assumptions about the data and the learner that build a model from that data. While such assumptions are possibly true, they cannot be checked for all the models' hidden way inside a model store. Hence, we will use non-parametric methods for our data extrapolation. Specifically, we will use the data manifolds seen in the leaf nodes of Figure 2b.

[4] `https://proceedings.neurips.cc/paper/2020/hash/d83de59e10227072a9c034ce10029c39-Abstract.html`

6 Intellectual Merit and Broader Impact

6.1 Intellectual Merit

Much prior work on unfairness detection and mitigation has made limiting assumptions about the nature of their work. For example, work on software fairness testing assumed extensive access to the model (and the data used to create it). In many scenarios, it may not be possible to grant such extensive access. Hence, we offer tools that support a broader class of problems; specifically: semi-supervised multi-goal learners that work, even if they can access only a small fraction of the data.

6.2 Broader Impact

We focus on issues of tremendous social importance—the detection and removal of unfair conclusions resulting from the application of AI tools. This work will increase America's ability for industrial and academic innovators to produce more AI-based software as well as certifying that those tools are fair to their users.

6.3 BPC Work: Broader Participation in Computer Science

PI Menzies will continue his established tradition of graduating research students for historically under-represented groups. NC State's Computer Science department has a strong record of studying research issues related to gender bias [89], barriers faced by women [43], and methods for broadening participation [68] in the context of software engineering. As to other impacts, this work will inform the curriculum of the various NC State NSF-funded REUs (research experience for undergraduates) as well as graduate SE classes (taught by PI Menzies).

Also, PI Menzies is a member of his department's Broadening Participation Committee (BPC) that actively seeks to:

- *Understand* what factors make computer science more (or less) attractive to underrepresented groups;
- *Educate* faculty, staff, and students on how different behaviors and activities effect diversity, quality and inclusiveness;
- *Increases* the percent of students who identify as women;
- *Evaluate* the success of his department's BPC team in broadening participating.

More immediately, this proposal will under two activities to broadening participation in computing (BPC) components:

- Tutorial notes and exercises on those procedures, to be presented at NCSU graduate classes and the NCSU summer schools;
- Funds from this work will be used to support students attending the Grace Hopper conference, and the Richard Tapia Celebration of Diversity in Computing.

As to other impacts, this work will inform the curriculum of the various NC State NSF-funded REUs(research experience for undergraduates) as well as graduate SE classes (taught by PI Menzies).

6.4 Dissemination of Knowledge

All the code developed as part of this work will be released as open-source software in GitHub, under an MIT license. Included in those packages will be the data used to certify the scripts as well as *RQn.sh* files containing executable scripts to reproduce (e.g.) RQ1.

Apart from that, the PI has a long history of publishing papers along with issue reproduction packages (complete with the data and the scripts required to replicate the papers' results). For a decade, PI Menzies lead-by-example in the open science community (the PROMISE project and the ROSE initiative) which takes care not only to package and distribute research code but also publish papers and tutorials on that material. In this regard, PI Menzies has an extensive record of successfully sharing his work with the community. For example, in June 2020, Google Scholar listed 50 papers as "most-cited" in the last five years at the IEEE Transactions on SE journal. Ten of those papers (20%) use data sets initially shared, the popularized, by PI Menzies.

7 Prior Results

PI Menzies is an IEEE Fellow and has earned over $13 million dollars in peer-reviewed competitive grants ($6.4M from NSF, and the rest for a variety of other government and industrial sources). Google Scholar lists him as a top-ten researcher in many research areas including knowledge acquisition and analytics. Serving as committee chair, he has graduated 12 Ph.D. and 32 masters students (by research). He currently supervises 10 Ph.D. students at NC State. He has served as an associated editor on all the major SE journals and from 2021 will be EIC of the Automated Software Engineering journal.

We include below some notes on some of his most recent NSF grants.

PI Menzies worked on (a) CCF-1302216, 2013-2107, $271,553; (b) "SHF: Medium: Collaborative: Transfer Learning in Software Engineering"; (c) The **intellectual merit** of that work was to define novel methods for sharing data, many of which were the precursor to the methods of this proposal. That work generated the publications (d) [44, 46, 60–63, 74, 81] concerning prediction and planning methods. The **broader impact** of that work was to enable a new kind of open science– one where all data is routinely shared and is capable of building effective models no matter if it is obfuscated for security purposes. The methods of this project, while targeted at software engineering, could also be applied to any other data-intensive field. (e) Data from that work is now housed in the two publicly accessible repository[5]. That work funded two Ph.D.s at NCSU. (f) N/A.

One related data mining grant is (a) OAC-1931425, 2019-2022, $592,129; (b) "Elements: Can Empirical SE be Adapted to Computational Science?"; (c) The **intellectual merit** of that work was to create a workbench containing methods adapted from empirical software engineering, that would help bridge the skill gap via automatic agents by suggesting to developers when they should investigate or redo part of their code. The **broader impact** is to reduce the associated cost (time, money, etc.) required to handle many of the large and more tedious aspects of software development. (d) That work generated one journal paper (at TSE), an MSR conference paper and two other papers under conference review [22, 90, 91]. (e) Data from that work is housed at the SEACRAFT publicly accessible repository [71]. That work funded one Ph.D. at NCSU. (f) N/A.

Another relevant research grant is (a) OAC-1826574, 2018-2018, $124,628.00; (b) "EAGER: Empirical Software Engineering for Computational Science"; (c) The **intellectual merit** of that work was to conduct initial explorations into novel methods for adapting SE methods to computational science. That work lead to the curious result that, in many ways, the computational scientists are better at managing their development cycle than many SE projects [90]. Whenever we found good enough data to compare the results seen in open source and computational science projects, we often find higher productivity values (and faster debugging) in computational science than in software engineering. (d) That work generated one journal paper (at TSE'21), one conference paper (at MSR'21) and another journal publication under review [67]. (e) Data from that work is now housed at the SEACRAFT publicly accessible repository [71]. That work funded one Ph.D. at NCSU. (f) N/A.

See also (a) CCF-1703487; 2017-2021, $898,349.00; (b) SHF: Medium: Scalable Holistic Autotuning for Software Analytics; (c) The **intellectual merit** of that work was that there exist previously unexplored "short-cuts" in the search space of control parameters of data miners. The **broader impact** was that better learners could be created automatically via "hyperparameter optimizers" that exploited those short-cuts. This in term meant that anywhere these learners were deployed, they could be deployed again with *greater* effect. (d) This work generated five journal articles at TSE, EMSE, IEEE Software, and one other article currently under review at EMSE [23, 72, 94, 96, 98, 99]; (e) Data from that work is now housed at the SEACRAFT publicly accessible repository [71]. That work funded two Ph.D. at NCSU. (f) N/A.

[5] `https://github.com/rshu/Adversarial-Evasion-Defense`, `https://github.com/rshu/e-dom`

References Cited

[1] "Amazon just showed us that 'unbiased' algorithms can be inadvertently racist." [Online]. Available: https://www.businessinsider.com/how-algorithms-can-be-racist-2016-4

[2] "Wolfram neural net repository." [Online]. Available: https://resources.wolframcloud.com/NeuralNetRepository

[3] "Uci:adult data set," 1994. [Online]. Available: http://mlr.cs.umass.edu/ml/datasets/Adult

[4] "Uci:statlog (german credit data) data set," 2000. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data)

[5] "Uci:heart disease data set," 2001. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Heart+Disease

[6] "Situation testing for employment discrimination in the united states of america," 2007. [Online]. Available: https://www.cairn.info/revue-horizons-strategiques-2007-3-page-17.htm

[7] "Proving discrimination cases – the role of situation testing," 2009. [Online]. Available: https://www.migpolgroup.com/_old/portfolio/proving-discrimination-cases-the-role-of-situation-testing/

[8] "Student performance data set," 2014. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Student+Performance

[9] "Medical expenditure panel survey," 2015. [Online]. Available: https://meps.ahrq.gov/mepsweb/

[10] "propublica/compas-analysis," 2015. [Online]. Available: https://github.com/propublica/compas-analysis

[11] "Machine bias," *www.propublica.org*, May 2016. [Online]. Available: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

[12] "Uci:default of credit card clients data set," 2016. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

[13] "Bank marketing uci," 2017. [Online]. Available: https://www.kaggle.com/c/bank-marketing-uci

[14] "Thome credit default risk," 2017. [Online]. Available: https://www.kaggle.com/c/home-credit-default-risk

[15] "Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias," 10 2018. [Online]. Available: https://github.com/IBM/AIF360

[16] "Amazon scraps secret ai recruiting tool that showed bias against women," Oct 2018. [Online]. Available: https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G

[17] "Ethics guidelines for trustworthy artificial intelligence." 2018. [Online]. Available: https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai

[18] "Ethically-aligned design: A vision for prioritizing human well-begin with autonomous and intelligence systems." 2019.

[19] "Microsoft ai principles. 2019." 2019. [Online]. Available: https://blogs.microsoft.com/eupolicy/artificial-intelligence-ethics/

[20] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2870052

[21] A. Adiga, C. J. Kuhlman, M. Marathe, S. Ravi, and A. Vullikanti, "Pac learnability of node functions in networked dynamical systems," in *International Conference on Machine Learning*, 2019, pp. 82–91.

[22] A. Agrawal and T. Menzies, "Is" better data" better than" better data miners"?" in *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*. IEEE, 2018, pp. 1050–1061.

[23] A. Agrawal, X. Yang, R. Agrawal, X. Shen, and T. Menzies, "Simpler hyperparameter optimization for software analytics: Why, how, when?" *IEEE Trans SE*, 2021.

[24] M. Amer, M. Goldstein, and S. Abdennadher, "Enhancing one-class support vector machines for unsupervised anomaly detection," 08 2013, pp. 8–15.

[25] A. A. Araújo, M. Paixao, I. Yeltsin, A. Dantas, and J. Souza, "An architecture based on interactive optimization and machine learning applied to the next release problem," *Automated Software Engineering*, vol. 24, no. 3, pp. 623–671, 2017.

[26] N. Beldiceanu and H. Simonis, "A model seeker: Extracting global constraint models from positive examples," in *Principles and Practice of Constraint Programming*, M. Milano, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 141–157.

[27] R. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilovic, S. Nagar, K. Natesan Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, R. Kush, and Y. Zhang, "Ai fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias," 10 2018.

[28] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth, "Fairness in criminal justice risk assessments: The state of the art," 2017.

[29] Y. Brun and A. Meliou, "Software fairness," in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2018, pp. 754–759.

[30] T. Calders and S. Verwer, "Three naive bayes approaches for discrimination-free classification," *Data Min. Knowl. Discov.*, vol. 21, no. 2, p. 277–292, Sep. 2010. [Online]. Available: https://doi.org/10.1007/s10618-010-0190-x

[31] F. Calmon, D. Wei, B. Vinzamuri, K. Natesan Ramamurthy, and K. R. Varshney, "Optimized pre-processing for discrimination prevention," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 3992–4001. [Online]. Available: http://papers.nips.cc/paper/6988-optimized-pre-processing-for-discrimination-prevention.pdf

[32] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" *arXiv preprint arXiv:2105.12195*, 2021.

[33] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies, "Fairway: A way to build fair ml software," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ser. ESEC/FSE 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 654–665. [Online]. Available: https://doi.org/10.1145/3368089.3409697

[34] 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

[35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, p. 321–357, Jun 2002. [Online]. Available: http://dx.doi.org/10.1613/jair.953

[36] N. Cowan, "The magical number 4 in short-term memory: a reconsideration of mental storage capacity," *Behav Brain Sci*, vol. 24, no. 1, pp. 87–114, Feb 2001.

[37] L. De Raedt, A. Passerini, and S. Teso, "Learning constraints from examples," in *32nd AAAI Conference on Artificial Intelligence/30th Innovative Applications of Artificial Intelligence Conference/8th AAAI Symposium on Educational Advances in Artificial Intelligence*. Assoc Advancement Artificial Intelligence, 2018, pp. 7965–7970.

[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, 2000.

[39] F. Denis, R. Gilleron, and F. Letouzey, "Learning from positive and unlabeled examples," *Theoretical Computer Science*, vol. 348, no. 1, pp. 70–83, 2005.

[40] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning," *Pattern Recogn.*, vol. 58, no. C, p. 121–134, Oct. 2016. [Online]. Available: https://doi.org/10.1016/j.patcog.2016.03.028

[41] C. Faloutsos and K.-I. Lin, "Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets," in *Proceedings of the 1995 ACM SIGMOD international conference on Management of data*, 1995, pp. 163–174.

[42] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," in *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 259–268.

[43] D. Ford, J. Smith, P. J. Guo, and C. Parnin, "Paradise unplugged: Identifying barriers for female participation on stack overflow," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2016, pp. 846–857.

[44] W. Fu, T. Menzies, and X. Shen, "Tuning for software analytics: Is it really necessary?" *Information and Software Technology*, vol. 76, pp. 135–146, 2016.

[45] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," 2016. [Online]. Available: https://arxiv.org/abs/1610.02413

[46] Z. He, F. Peters, T. Menzies, and Y. Yang, "Learning from open-source projects: An empirical study on defect prediction," in *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on*. IEEE, 2013, pp. 45–54.

[47] J. Hui, "Machine learning —expectation-maximization algorithm (em)," 2019. [Online]. Available: https://jonathan-hui.medium.com/machine-learning-expectation-maximization-algorithm-em-2e954cb76959

[48] T. Ishida, G. Niu, and M. Sugiyama, "Binary classification from positive-confidence data," in *Advances in Neural Information Processing Systems*, 2018, pp. 5917–5928.

[49] N. Japkowicz, C. Myers, and M. Gluck, "A novelty detection approach to classification," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, p. 518–523.

[50] D. Ji, P. Smyth, and M. Steyvers, "Can i trust my fairness metric? assessing fairness with unlabeled data and bayesian inference," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/d83de59e10227072a9c034ce10029c39-Abstract.html

[51] B. Johnson, J. Bartola, R. Angell, K. Keith, S. Witty, S. J. Giguere, and Y. Brun, "Fairkit, fairkit, on the wall, who's the fairest of them all? supporting data scientists in training fair models," *arXiv preprint arXiv:2012.09951*, 2020.

[52] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," *Knowledge and Information Systems*, vol. 33, no. 1, pp. 1–33, Oct 2012. [Online]. Available: https://doi.org/10.1007/s10115-011-0463-8

[53] F. Kamiran, S. Mansha, A. Karim, and X. Zhang, "Exploiting reject option in classification for social discrimination control," *Inf. Sci.*, vol. 425, no. C, p. 18–33, Jan. 2018. [Online]. Available: https://doi.org/10.1016/j.ins.2017.09.064

[54] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, "Fairness-aware classifier with prejudice remover regularizer," in *Machine Learning and Knowledge Discovery in Databases*, P. A. Flach, T. De Bie, and N. Cristianini, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 35–50.

[55] R. Kerber, "Chimerge: Discretization of numeric attributes," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, ser. AAAI'92. AAAI Press, 1992, p. 123–128.

[56] N. S. S. A. T. M. Kewen Peng, Christian Kaltenecker, "Veer: Disagreement-free multi-objective configuration," in *Submitted to ASE'21*, 2021.

[57] S. S. Khan and M. G. Madden, "A survey of recent trends in one class classification," in *Irish conference on artificial intelligence and cognitive science*. Springer, 2009, pp. 188–197.

[58] ——, "One-class classification: taxonomy of study and review of techniques," *The Knowledge Engineering Review*, vol. 29, no. 3, pp. 345–374, 2014.

[59] E. Kocaguneli, T. Zimmermann, C. Bird, N. Nagappan, and T. Menzies, "Distributed development considered harmful?" in *35th International Conference on Software Engineering, ICSE '13, San Francisco, CA, USA, May 18-26, 2013*, D. Notkin, B. H. C. Cheng, and K. Pohl, Eds. IEEE Computer Society, 2013, pp. 882–890. [Online]. Available: https://doi.org/10.1109/ICSE.2013.6606637

[60] R. Krishna, V. Nair, P. Jamshidi, and T. Menzies, "Whence to learn? transferring knowledge in configurable systems using beetle," *TSE*, 2020.

[61] R. Krishna and T. Menzies, "Learning effective changes for software projects," *arXiv preprint arXiv:1708.05442. Under review; IEEE TSE*, 2017.

[62] ——, "Bellwethers: A baseline method for transfer learning," *IEEE Transactions on Software Engineering*, vol. 45, no. 11, pp. 1081–1105, 2018.

[63] R. Krishna, T. Menzies, and W. Fu, "Too much automation? the bellwether effect and its implications for transfer learning," in *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering, ASE 2016, Singapore, September 3-7, 2016*, 2016, pp. 122–131. [Online]. Available: http://doi.acm.org/10.1145/2970276.2970339

[64] J. Larkin, J. McDermott, D. P. Simon, and H. A. Simon, "Expert and novice performance in solving physics problems," *Science*, vol. 208, no. 4450, pp. 1335–1342, 1980. [Online]. Available: http://science.sciencemag.org/content/208/4450/1335

[65] W. S. Lee and B. Liu, "Learning with positive and unlabeled examples using weighted logistic regression," in *ICML*, vol. 3, 2003, pp. 448–455.

[66] E. Levina and P. Bickel, "Maximum likelihood estimation of intrinsic dimension," in *Advances in Neural Information Processing Systems*, L. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17. MIT Press, 2005. [Online]. Available: https://proceedings.neurips.cc/paper/2004/file/74934548253bcab8490ebd74afed7031-Paper.pdf

[67] X. Ling, R. Agrawal, and T. Menzies, "How different is test case prioritization for open and closed source projects," *IEEE Transactions on Software Engineering*, mar 2021.

[68] F. Liu, D. Ford, C. Parnin, and L. Dabbish, "Selfies as social movements: Influences on participation and perceived impact on stereotypes," *Proc. ACM Hum.-Comput. Interact.*, no. CSCW, 2017. [Online]. Available: https://doi.org/10.1145/3134707

[69] A. Lustosa and T. Menzies, "Preference discovery in large product lines," *CoRR*, vol. abs/2106.03792, 2021. [Online]. Available: https://arxiv.org/abs/2106.03792

[70] W. J. Ma, M. Husain, and P. M. Bays, "Changing concepts of working memory," *Nature neuroscience*, vol. 17, no. 3, pp. 347–356, 2014.

[71] T. Menzies, R. Krishna, and D. Pryor, "The seacraft repository of empirical software engineering data," *Retrieved March*, 2017.

[72] T. Menzies, "Shockingly simple:" keys" for better ai for se," *IEEE Software*, vol. 38, no. 2, pp. 114–118, 2021.

[73] T. Menzies, A. Butcher, D. R. Cok, A. Marcus, L. Layman, F. Shull, B. Turhan, and T. Zimmermann, "Local versus global lessons for defect prediction and effort estimation," *IEEE Trans. Software Eng.*, vol. 39, no. 6, pp. 822–834, 2013. [Online]. Available: https://doi.org/10.1109/TSE.2012.83

[74] T. Menzies, W. Nichols, F. Shull, and L. Layman, "Are delayed issues harder to resolve? revisiting cost-to-fix of defects throughout the lifecycle," *Empirical Software Engineering*, vol. 22, no. 4, pp. 1903–1935, 2017.

[75] T. Menzies and T. Zimmermann, "Goldfish bowl panel: Software development analytics," in *34th International Conference on Software Engineering, ICSE 2012, June 2-9, 2012, Zurich, Switzerland*, M. Glinz, G. C. Murphy, and M. Pezzè, Eds. IEEE Computer Society, 2012, pp. 1032–1033. [Online]. Available: https://doi.org/10.1109/ICSE.2012.6227117

[76] ——, "Software analytics: So what?" *IEEE Softw.*, vol. 30, no. 4, pp. 31–37, 2013. [Online]. Available: https://doi.org/10.1109/MS.2013.86

[77] ——, "Software analytics: What's next?" *IEEE Softw.*, vol. 35, no. 5, pp. 64–70, 2018. [Online]. Available: https://doi.org/10.1109/MS.2018.290111035

[78] G. A. Miller, "The magical number seven, plus or minus two: some limits on our capacity for processing information." *Psychological review*, vol. 63, no. 2, p. 81, 1956.

[79] V. Nair, Z. Yu, T. Menzies, N. Siegmund, and S. Apel, "Finding faster configurations using flash," *TSE*, 2018.

[80] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5450–5463, 2019.

[81] F. Peters, T. Menzies, and L. Layman, "Lace2: Better privacy-preserving data sharing for cross project defect prediction," in *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 1. IEEE, 2015, pp. 801–811.

[82] S. Rajasegarar, C. Leckie, J. C. Bezdek, and M. Palaniswami, "Centered hyperspherical and hyperellipsoidal one-class support vector machines for anomaly detection in sensor networks," *Trans. Info. For. Sec.*, 2010.

[83] 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

[84] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Machine Intelligence*, vol. 1, no. 5, pp. 206–215, 2019. [Online]. Available: https://doi.org/10.1038/s42256-019-0048-x

[85] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," ser. Proceedings of Machine Learning Research, 2018.

[86] A. S. Sayyad, T. Menzies, and H. Ammar, "On the value of user preferences in search-based software engineering: A case study in software product lines," in *2013 35Th international conference on software engineering (ICSE)*. IEEE, 2013, pp. 492–501.

[87] 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.

[88] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, 1997.

[89] J. Terrell, A. Kofink, J. Middleton, C. Rainear, E. Murphy-Hill, C. Parnin, and J. Stallrich, "Gender differences and bias in open source: Pull request acceptance of women versus men," *PeerJ Computer Science*, 2017.

[90] H. Tu, R. Agrawal, and T. Menzies, "The changing nature of computational science software," *arXiv preprint arXiv:2003.05922*, 2020.

[91] H. Tu, G. Papadimitriou, M. Kiran, C. Wang, A. Mandal, E. Deelman, and T. Menzies, "Mining scientific workflows for anomalous data transfers," in *MSR'21*, 2021.

[92] G. Vilone and L. Longo, "Explainable artificial intelligence: a systematic review," *arXiv preprint arXiv:2006.00093*, 2020.

[93] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *EMO*, 2006.

[94] T. Xia, R. Shu, X. Shen, and T. Menzies, "Sequential model optimization for software process control," *IEEE TSE, to appear*, 2021.

[95] M. Xiu, Z. Ming, Jiang, and B. Adams, "An exploratory study on machine learning model stores," *IEEE Software*, 2021.

[96] X. Yang, J. Chen, R. Yedida, Z. Yu, and T. Menzies, "Learning to recognize actionable static code warnings (is intrinsically easy)," *Empir. Softw. Eng.*, vol. 26, no. 3, p. 56, 2021. [Online]. Available: https://doi.org/10.1007/s10664-021-09948-6

[97] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, ser. ACL '95. USA: Association for Computational Linguistics, 1995, p. 189–196. [Online]. Available: https://doi.org/10.3115/981658.981684

[98] R. Yedida and T. Menzies, "On the value of oversampling for deep learning in software defect prediction," *IEEE Transactions on Software Engineering*, pp. 1–1, 2021.

[99] R. Yedida, X. Yang, and T. Menzies, "When simple is better than complex: A case study on deep learning for predicting bugzilla issue close time," 2021.

[100] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," 2018. [Online]. Available: http://www.aies-conference.com/wp-content/papers/main/AIES_2018_paper_162.pdf

[101] T. Zhang, J. Li, M. Han, W. Zhou, P. Yu *et al.*, "Fairness in semi-supervised learning: Unlabeled data help to reduce discrimination," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[102] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in Neural Information Processing Systems*, S. Thrun, L. Saul, and B. Schölkopf, Eds., vol. 16. MIT Press, 2004. [Online]. Available: https://proceedings.neurips.cc/paper/2003/file/87682805257e619d49b8e0dfdc14affa-Paper.pdf

[103] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Tech. Rep., 2002.

[104] X. J. Zhu, "Semi-supervised learning literature survey," 2005.

[105] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2004, pp. 832–842.

[106] M. Zuluaga, A. Krause, and M. Püschel, "e-pal: An active learning approach to the multi-objective optimization problem," *Journal of Machine Learning Research*, vol. 17, no. 104, pp. 1–32, 2016. [Online]. Available: http://jmlr.org/papers/v17/15-047.html

Facilities, Equipment, and Other Resources

Facilities From other Institutions

As stated in our *Collaboration Plan*, we have unpaid collaborators from Facebook, Microsoft, and IBM. Once this research develops viable unfairness measurement and mitigation methods for semi-supervised algorithms, these collaborators will grant us access to materials, to be negotiated, at their site. Apart from (potentially) being able to access models behind the firewalls of model stores, an important facility these collaborators could offer is access to the subject matter experts that can test if the veracity of our artificially generated models.

Facilities at North Carolina State University

Offices:

The project PI has and offices in their CS Department. This department has adequate space to house all research assistants working on this project. All offices are wired for high-speed network access.

The PI's departments at NC State provide the space and basic networking services to carry out the experiments, secretarial and administrative support as well as general-purpose office equipment (*e.g.*, fax, photocopiers, etc.).

Lab Space

The PI has their own lab space at NC State. PI Menzies' RAISE lab (Real-World AI and SE) is a newly renovated space containing over 1,500 ft^2 of research space and 15 cubicles, a meeting space, printer, and wide screen projector.

Compute Facilities

Part of *xPLAIN* will involve comparatively assessing different technologies. For that process, it will be useful to have some large-scale compute facility.

At NCSU, students working on this grant will have access to a 108-node compute cluster named ARC with 2,000 cores (AMD Mangy-Cours), Infiniband QDR interconnect, per node power monitoring, GPUs and SSDs and parallel file system support, which was funded by an NSF CRI that he is the main PI of together with 5 co-PIs. The ARC facility is providing local and remote researchers with administrator/root privileges for Computer Science experiments at a medium scale. This allows any of the software layers, including the operating system and Infiniband switch network routing tables, to be modified for experimental purposes, e.g., to experiment with different network topologies. For large-scale demonstrations, other facilities will be utilized (the HPC discussed below.

Additionally, NC State University provides a High-Performance Computing (HPC) facility as a part of the initiative to provide state-of-the-art support for research and academic computing. HPC system (called henry2) provides NC State students and faculty with entry and medium-level high-performance research and education computing facilities, consulting support and scientific workflow support. The HPC ecosystem consists of 1233 dual Xeon compute nodes in the henry2 cluster. Each node has two Xeon processors (mix of dual-, quad-, six-, eight-, ten-core, twelve-core) and 2 to 6 GigaBytes of memory per core. The total number of cores increases as more cores are purchased and now exceeds 10000. The nodes all have 64-bit processors. All HPC projects have the capability to run jobs using up to 128 processor cores for up to 48 hours and smaller jobs up to a week.

Data Management Plan

Types of Data This research will produce public source code; example structured data sets; predictive models; predictive model's intermediate tuning cache ;records of results of applying the code to the data; and work-processing files (the reports of our results).

Standards for data and metadata format All data will be stored in different format, appropriate for the type of data being stored. For example, appropriate data formats for papers include word files, latex files, and PDF files. As another example, appropriate data formats for repository data including Attribute Relation File Format (.arff) or Comma-Separated Values (.csv). Similarly Hierarchical Data Format (.h5) or Pickle(.pkl) for storing generated models.

All our scripts used here will be based around software tools in widespread use (GitHub, Travis CI, etc.). We will release all our tools via open source licenses, so our results can be readily applicable to researcher or developers using GitHub, etc.

Policies for access and sharing Information identifying individuals will not be shared. As to everything else, where possible, open source data and software will be shared public through repository (with an exception for qualitative survey results which may be presented as survey summary after applying de-identification of the responses from individuals). Repository data will be freely available to the world and licensed under a Creative Commons Attribution 4.0. International license (https://creativecommons.org/licenses/by/4.0/). This data will be made freely available our repository, accessible through the internet.

Policies and provisions for re-use, re-distribution, and derivatives All published and open source data may be re-used, re-distributed, and derived as long as the original data is not misrepresented, and the materials do not violate the acceptable use of the IP holders. Others may change or alter the open source data for personal purposed provided that these changes are made clear in any publications, re- distributions, or derivations.

Plans for archiving data and other research products Repository data will be hosted on Zenodo repository (hosted at the Large Hadron Collider in Switzerland). Papers will be published at major conference and journals in SE data analytics.

1. Tim Menzies, North Carolina State University; PI

Collaboration Plan

Once this research develops viable unfairness measurement and mitigation methods for semi-supervised algorithms, we have industrial collaborators willing to grant us access to materials at their site.

It would be premature to negotiate the precise details of that collaboration until we have running prototypes for the RQ1 and RQ2 work (which we estimate will be available sometime in mid 2024). At this time, what we can say is that all the following researchers have expressed interest in this work and would, in 2024, be willing to review our prototypes, then explore options for unpaid collaboration. Apart from (potentially) being able to access models behind the firewalls of model stores, an important facility these collaborators could offer is access to the subject matter experts that can test the veracity of our artificially generated models.

Our *Supplemental Documents* section contain letters of collaboration from Microsoft and Facebook. Our IBM colleagues asked to see the RQ1,RQ2 results before writing letters.

Dr. Thomas Zimmermann (IEEE Fellow; Senior Principal Researcher at Microsoft Research; Co-Editor In Chief of the Empirical Software Engineering Journal; Chair of ACM SIGSOFT; ACM Distinguished Scientist.)
- https://thomas-zimmermann.com/
- tzimmer@microsoft.com
- Dr. Zimmermann explores a diverse portfolio of softare analytics research at Microsoft.
- He has published extensively on biases in software engineering and how they can effect productivity.
- Dr. Zimmermann and PI Menzies have a long history of productive collaborative research and have published research results at IEEE Transactions on Software Engineering [73], the International Conference on Software Engineering [59], the Automated SE conference [75], and IEEE Software [76,77].
- For this part of the collaboration, we would check if our tools can find the same (or different) biases to those previously reported.

Dr. Erik Meijer (Facebook Director Of Engineering)
- https://www.linkedin.com/in/erikmeijer1
- erikm@fb.com
- Dr. Meijer is software leader at Facebook.
- For this part of the collaboration, we would check the models Facebook uses to manage that very large developer team (and check that they are avoiding discriminatory practices).

Dr. Rachel Bellamy (Principal Researcher Thomas J. Watson Research Center, Yorktown Heights, NY USA).
- https://researcher.watson.ibm.com/researcher/view.php?person=us-rachel;
- rachel@us.ibm.com
- In her current role. Dr/.Bellamy heads the council that curates and manages the IBM Research portfolio of exploratory science projects.
- Dr Bellamy is a leader in the team that built the IBM AI360 fairness toolkit [15], which is a widely used fairness mitigation tools. AI360 is another example of a fairness toolkit that requires extensive access to the data. Note that if we can add xPLAIN into AI360, then that would greatly extend the industrial population of users that take advantage of our tools.
- For this collaboration, we would check if our "data-lite" methods can augment the AI360 tools.

Dr Maja Vuković (IBM Fellow, responsible for technical and research strategy for AI driven Application Modernization)
- https://researcher.watson.ibm.com/researcher/view.php?person=us-maja
- maja@us.ibm.com
- Dr. Vuković works on cloud infrastructure at the Thomas J. Watson Research Center, Yorktown Heights, NY USA.
- Dr. Vuković and Dr. Menzies have been working closely together this last year on software configuration for cloud applications.
- In this part of the collaboration, we would assess what extra requirements are needed inside the firewalls of a model store if it was determined that all models should get tested for fairness.