



How fair can we go in machine learning?

Assessing the boundaries of accuracy and fairness

Ana Valdivia¹ | Javier Sánchez-Monedero² | Jorge Casillas³

¹Department of War Studies, King's College London, London, UK

²School of Journalism, Media and Culture, Data Justice Lab, Cardiff University, Cardiff, UK

³Department of Computer Science and Artificial Intelligence, Data Science and Computational Intelligence Institute (DaSCI), University of Granada, Granada, Spain

Correspondence

Ana Valdivia, Department of War Studies, King's College London, London WC2R 2SL, UK.

Email: ana.valdivia@kcl.ac.uk

Funding information

European Research Council, Grant/Award Number: 759903; European Union's Horizon 2020 research and innovation programme, Grant/Award Number: 819213; Spanish Ministry of Science and Innovation, Institute of Health Carlos III, Grant/Award Number: PI20/01435

Abstract

Fair machine learning has been focusing on the development of equitable algorithms that address discrimination. Yet, many of these fairness-aware approaches aim to obtain a unique solution to the problem, which leads to a poor understanding of the statistical limits of bias mitigation interventions. In this study, a novel methodology is presented to explore the tradeoff in terms of a Pareto front between accuracy and fairness. To this end, we propose a multiobjective framework that seeks to optimize both measures. The experimental framework is focused on logistiregression and decision tree classifiers since they are well-known by the machine learning community. We conclude experimentally that our method can optimize classifiers by being fairer with a small cost on the classification accuracy. We believe that our contribution will help stakeholders of sociotechnical systems to assess how far they can go being fair and accurate, thus serving in the support of enhanced decision making where machine learning is used.

KEYWORDS

algorithmic fairness, group fairness, multiobjective optimization

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2020 The Authors. *International Journal of Intelligent Systems* published by Wiley Periodicals LLC

1 | INTRODUCTION

Algorithmic and data-driven decision making has rapidly swept through several social, political, and industry contexts. Beyond the possible misuses of technology, there is an increased awareness that these processes are not neutral and can reproduce and amplify past and current structural inequalities.^{1,2} Within this context, particular interest is paid to the role of machine learning (ML) with well known examples of models biased against historically discriminated groups³⁻⁵ or the intersection of these groups.^{6,7} Fairness, Accountability, Transparency, and Ethics (FATE) in ML has emerged as a community initially motivated to develop technological solutions to the disparate impact and treatment by biased algorithms^{5,8-11} that also moves to a broader and multidisciplinary understanding of the issues of sociotechnological interventions.¹²⁻¹⁵ The present work contributes to this field by studying how far bias mitigation can go while satisfying the accuracy of the models, providing a tool for a wider understanding of accuracy and fairness tradeoff.

Bias mitigation techniques can broadly be divided into three nonexclusive categories¹⁶: (1) preprocessing, (2) inprocessing, and (3) postprocessing. The preprocessing techniques attempt to learn new representations of data to satisfy fairness definitions. The inprocessing methods involve modifying the classifier algorithm by adding a fairness criteria to the optimization problem. The postprocessing methods aim at removing discriminatory decisions after the model is trained. Normally, in inprocessing approaches the fairness criteria are used as an optimization constraint rather than as a guide to build a more equitable prediction model. As a result of the optimization process, those fixed restrictions will come out with a degree of equity that might not match the problem requirements whereas the space of solutions that can be reached remains unknown so that decision makers cannot observe the range of possibilities and their behavior.

The main contribution of this paper is a methodology that explores optimal ML solutions and evaluates the boundaries of fairness in relation to other dimensions of the evaluation of an ML model. We claim that multiobjective evolutionary algorithms might be used to direct a meta-learning process for optimizing the hyperparameters of a classifier. Thus, we propose to use a genetic algorithm to tune learner hyperparameters to find models that offer a wide repertoire of balances between precision and fairness. The architecture of this methodology can be applied to any type of classifier and hyperparameter set and the optimization is independent of the definition of fairness and precision. In particular, we focus the study on the suitability of both logistic regression and decision trees as base learners because of their properties of good accuracy with considerable simplicity (in the former case) and transparency (in the latter case). As a result of the meta-learning process, the method produces a Pareto front with a set of suboptimal feasible solutions. In this way, the method addresses the previous issues of single constrained optimization proposals to build fair models.

We conduct an extensive set of experiments based on five real-world data sets which are widely used in the FATE literature. The solution space obtained by our approach indicates that there exists a wide number of optimal solutions (Pareto optimal) that are characterized by not being dominated by each other. We also evaluate the boundaries between accuracy and fairness that can be achieved on each problem, giving an empirical visualization of the limits between both measures. In addition, we assess how decision trees hyperparameters are affected by this tradeoff. Finally, a convergence analysis is also presented to evaluate the evolutionary methodology.

As far as we know, multiobjective optimization has not yet been used in the field of FATE in ML, so we believe that the proposal will open a very fruitful and beneficial research line, enriching the current state-of-the-art.

2 | BACKGROUND

To ground our methodology, we begin by reviewing relevant works in bias mitigation (Section 2.1). We then introduce evolutionary algorithms for multiobjective optimization (Section 2.2).

2.1 | Optimizing fairness and accuracy

Bias mitigation algorithms often explicitly or implicitly add fairness constraints on model group performance. Typically there is a categorical binary variable for group membership which is often referred as sensitive attribute. In this section, we introduce some related works that aim at optimizing for fairness and accuracy. For further information on the relation between accuracy and fairness measures, we refer to Reference [17].

Logistic regression algorithms have been widely used in fairness literature. For instance, the authors in Reference [8] presented a flexible convex optimization framework that minimizes the accuracy loss function subject to fairness constraints. The method is valid for boundary-based classifiers such as logistic regression and proved that it allows to control fairness, often at a small cost in accuracy. In the context of decision trees, in Reference [18] the information gain function used for splitting and pruning is modified to add the entropy with respect to the sensitive attribute as splitting or pruning criteria. The authors explored several options. The first one considers the entropy with respect to the class label, but it does not allow splitting if it introduces discrimination with respect to sensitive attribute. The second alternative implements a tradeoff between objectives by dividing the gain in accuracy by the gain in discrimination however this option did not achieve suitable results.

More recently, authors in Reference [19] proposed to reduce fair classification to a sequence of cost-sensitive classification tasks to obtain Pareto optimality between overall accuracy and any fairness definition. In related work,²⁰ Balashankar et al. found a Pareto optimal point which maximizes multiple subgroup accuracy measures while satisfying equality of opportunity.

Zafar et al.²¹ formulated the problem as a convex constrained optimization problem that allows a dual formulation in which accuracy is optimized under fairness constraints. In their formulation, fairness is introduced in terms of a measure of the fairness of the decision boundary that serves as a proxy to many fairness statistical metrics. The tradeoff between accuracy and fairness due to disparate mistreatment is expressed as a threshold parameter established by the user. Moreover, the formulation allows introducing several attributes as constraints, for example, race and gender.

Hu et al.²² transformed the constrained loss minimization problem into a social welfare maximization problem. Using support vector machines regularization path and techniques from parametric programming, they showed that always preferring more fair solutions does not abide by the Pareto Principle. They concluded that applying strict fairness criteria can lead to worse welfare outcomes for the groups.

This brief literature review reveals the research interest in exploring the simultaneous optimization of accuracy and fairness. While some proposals obtain Pareto optimal solutions that implicitly set a tradeoff between objectives, other works introduce a user parameter to explicitly define the tradeoff. As an alternative to this, our work aims to provide the whole Pareto front as a means to train learners and to explore the impact of the models, and, in

general, to better understand the behavior of the combination between a data set and knowledge representation.

2.2 | Multiobjective evolutionary algorithms

Multiobjective optimization is a field of decision making which aims at optimizing simultaneously more than one objective function. This field of research has developed a large number of applications in engineering, economics, and logistics, where optimal decisions need to be taken in the presence of tradeoffs between two or more competitive objectives. Maximizing comfort and energy saving in a climatization system is a practical example of multiobjective problem involving two objectives. Mathematically, this can be formulated as follows:

$$\min(f_1(x), \dots, f_n(x)) \quad \text{s. t. } x \in X,$$

where $n > 1$ is the number of objective functions and X is the set of feasible solutions.

When multiple objective functions appear in a problem, no single solution exists that optimizes each function at once. Otherwise, the presence of multiple objectives gives a set of optimal solutions, possibly infinite. A solution is *nondominated* whether does not exist another solution that dominates the current one, that is, it does not improve one objective function without worsening other objective functions.

Definition 2.1. A solution $x \in X$ is said to *dominate* another solution $x' \in X$, if it is better or equal in all the objectives and strictly better in at least one of them, that is,

- $f_i(x) \leq f_i(x'), \forall i \in \{1, \dots, n\}$ and,
- $f_j(x) < f_j(x')$, for at least one index $j \in \{1, \dots, n\}$.

A solution is called *Pareto optimal* if there does not exist another solution that dominates it. Consequently, the set of all Pareto optimal solutions is defined as *Pareto front* or *boundary*. Assessing this frontier allows decision-makers to select any efficient solution, depending on the worthiness of each objective function.

An evolutionary algorithm is a family of bioinspired meta-heuristic algorithms which often are well-suited for solving optimization problems. Inspired by some aspects of natural evolution, the basic idea is that fitter individual, this is the solutions to a problem, are more likely to survive and thus contribute to the gene pool of the offspring while unfit members will not likely contribute to the following generations. Over the last decades, a number of multiobjective evolutionary algorithms have been developed to search for multiple Pareto optimal solutions.

3 | MULTIOBJECTIVE METHOD FOR ACCURATE AND FAIR MACHINE LEARNING

We propose a methodology based on the nondominated sorting genetic algorithm II (NSGA-II) (see Appendix A) to train a set of classifiers that best tradeoff accuracy and fairness. To obtain the Pareto optimal solutions, the meta-heuristic algorithm will optimize the combination of learner parameters. The selection mechanisms are inspired by the elitist NSGA-II method²³

which was described in the previous section. As proof of concept, we tested our methodology with logistic regression and decision trees as base ML classifiers.

3.1 | Meta-learning approach

The pseudo-code of the meta-learning approach is presented in Algorithm 1. Additionally, Figure 1 presents a visual diagram of the workflow.

Algorithm 1: Meta-learning algorithm

Input: objective function of accuracy and fairness (f_1 and f_2), number of hyperparameters of the ML classifier (m), intervals of hyperparameters ($\min(h_i)$ and $\max(h_i) \forall i \in \{1, \dots, m\}$), data sets, and the protected attribute

Output: Set of ML models with different accuracy-fairness tradeoffs

Data: training (learning and validation) and testing data set (D_{learn} , D_{val} , and D_{test})

Parameters: number of generations (G), population size (N), crossover probability (p_c), mutation probability (p_m), mutation parameter (μ)

begin

 initialize population P_1

 evaluate objective functions (P_1 , D_{val})

 nondominated rank individuals of P_1

while $k \leq G$ **do**

$P_k^{(1)} \leftarrow$ elitist selection (P_{k-1})

$P_k^{(2)} \leftarrow$ crossover ($P_k^{(1)}$)

$P_k^{(3)} \leftarrow$ mutation ($P_k^{(2)}$)

while $1 \leq l \leq N$ **do**

 create S_{kl} solution by training classifier (I_{kl} , D_{learn})

 evaluate objective functions (S_{kl} , D_{val})

end while

 nondominated rank individuals of population $P_k^{(3)}$

$P_k \leftarrow$ elitist nondominated replacement ($P_k^{(3)}$, P_{k-1})

end while

return nondominated solutions in P_k

end

Specifically, the meta-learning consists of dividing the training set into two subsets (learning and validation) where the classification models will be built from the first set, and fairness and accuracy will be measured with the second one. The multiobjective algorithm will ensure that, in each iteration, the set of the best hyperparameter configurations will survive so that the NSGA-II will explore new settings around them. At the end of the meta-learning process, a set of suboptimal solutions is returned and evaluated with the testing set.

The main advantage of the proposed method is that it can obtain a wide number of suboptimal solutions in one run. Also, the method allows to use any ML classifier without modifying it as unlike the inprocessing technique. These learner-dependent components (coding scheme and initialization) are described in next Section 3.2, while subsequent Sections 3.3–3.5 extensively describe the rest of evolutionary algorithm components that are independent of the base learner used.

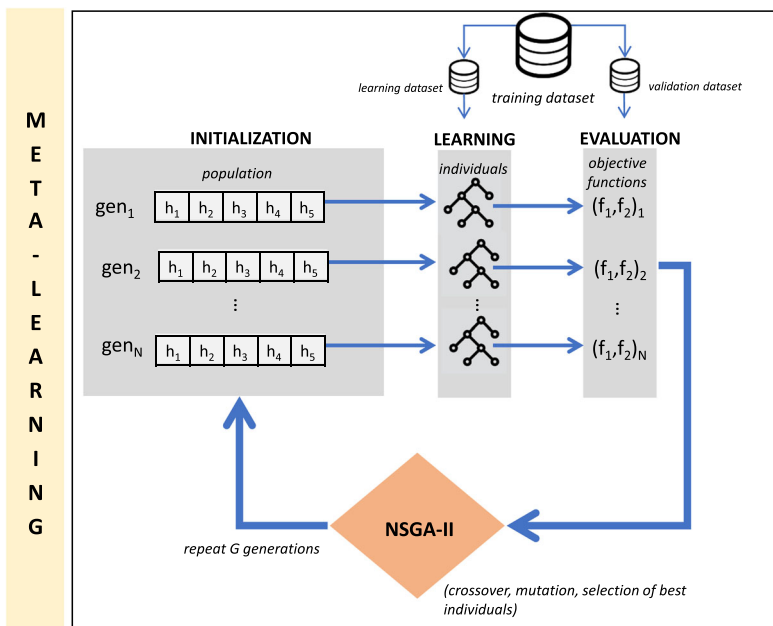


FIGURE 1 This diagram overviews the flow of the proposed meta-learning. The first population is randomly generated at the initialization step. Given the values of each gene, N learners are trained and evaluated with each combination of hyperparameters afterwards. The NSGA-II ranks the individuals, that is the trained learners, by evaluating the objective functions on the validation set. After that, the NSGA-II generates an offspring population which is also evaluated. Finally, the method selects the best N -members among parents and children to form the next population using a selection mechanism known as *elitist nondominated sorting*. This process is repeated until the last generation G is reached [Color figure can be viewed at wileyonlinelibrary.com]

3.2 | Coding scheme and pool initialization: Machine learning classifiers

The coding scheme and the pool initialization are the two only components of the proposed method that depends on the base machine learner. In this section we explain the coding scheme and initialization for logistic regression and decision trees.

3.2.1 | Logistic regression

Classifier

Logistic regression also referred to as logit, is considered one of the most used learning methods for classification. This classifier is a very transparent and intelligible model, it fits a linear equation that predicts an outcome for a binary variable. However, the input data needs to be standardized to properly interpret the coefficients and the relationship between the input and output.

One important concept related with logistic regression is *regularization*. Any modification of a learning method to improve performance on the unseen data sets is called regularization. Generally, in the logistic regression model a penalty term is added to the loss function, which is known as the l2 penalty.

Hyperparameters

We have considered the following hyperparameters of the logistic regression:

- `max_iter`: Maximum number of iterations taken for the solvers to converge with values in [20,200].
- `tol`: Tolerance for stopping criteria taking values in [1e-4,1e-1].
- `lambda`: Regularization parameter taking values in [1e-3,1e5].
- `l1_ratio`: It is used to specify a combination of penalties L1 and L2. The parameter takes values in [0,1], with 0 meaning that only L2 is used, while 1 means only L1 is used.
- `class_weight`: It is used to give weight to each class, which is considered when measuring the quality of the splits. It is very useful for unbalanced datasets where models usually misclassified the minority class. It takes values in [0, 1]. The positive class is weighted with `class_weight`, while the negative one is `1-class_weight`. A value of 0.5 means both classes are evenly considered.

Using the logistic regression learner, the j th-individual, I_{kj} , of the k th-population, P_k is a trained model. In turn, this model is trained with a m -tuple gen, g_{kj} , which contains the values of each hyperparameter $h = \{h_1, ..., h_m\}$ on each corresponding position, hence $m = 5$:

$$I_{kj} := \text{logistic_regression}(g_{kj})$$

$$h := \{\text{max_iter}, \text{tol}, \text{lambda}, \text{l1_ratio}, \text{class_weight}\}.$$

Pool initialization

The initialization step generates the first pool. The first individual generated (I_{11}) is created with the following default values of hyperparameters: $g_{11} = (100, 0.0001, 1, 0, \text{None})$. The second individual (I_{12}) is created with the same values but using L1 norm: $g_{11} = (100, 0.0001, 1, 1, \text{None})$. The remaining individuals are generated at random within the variation interval of each hyperparameter.

3.2.2 | Decision tree

Classifier

Decision trees are considered *white box* models, since it is easy to analyze the steps taken to classify data.²⁴ They are easy to interpret, and they can be summarized in a set of rules. This fact supports another of the FATE community's claims, which is transparency. By using decision trees as classifiers, we allow decision makers to understand the behavior of the model.

In addition, these kind of algorithms do not require data normalization or dummy variables creation, since they are able to use both numerical and categorical data. This fact simplifies the preprocessing step, which can directly affect the accuracy and fairness of the classifier.¹⁶

Hyperparameters

We consider the following hyperparameters of the decision tree learner:

- `criterion`: This function measures the quality of a split. Decision trees split nodes as long as this value decreases. The purity of a node can be measured with the *Gini* index and the *entropy*.
- `max_depth`: The maximum depth of the tree. Deeper trees are more complex.

- `min_samples_split`: The minimum number of samples required to divide an internal node. In this case, a higher number of samples tends to produce simpler trees.
- `max_leaf_nodes`: Total number of leaves in a tree. The higher the number of leaves, the more complex the tree.
- `class_weight`: Same as before (See hyperparameters of logistic regression).

The `criterion`, `max_depth`, and `min_samples_split` adjust the size of the tree in different directions, which means that different balances between precision and complexity can be found. Moreover, if the search of the best set of hyperparameters is guided by any fairness metric, the structure of the tree can be regulated towards branches that do not generate disparities among groups. The `class_weight` hyperparameter addresses disparity by transferring instances between false-positives and false-negatives.

In this case, the j th-individual, I_{kj} , of the k th-population, P_k , is a trained decision tree. In turn, this tree is trained with a m -tuple gen , g_{kj} , which contains the values of each hyperparameter $h = \{h_1, \dots, h_m\}$ on each corresponding position, hence $m = 5$:

$$I_{kj} := \text{decision_tree}(g_{kj})$$

$$h := \{\text{criterion}, \text{max_depth}, \text{min_samples_split}, \text{max_leaf_nodes}, \text{class_weight}\}.$$

Since three of the hyperparameters are categorical or integer numbers, those genes are rounded after their decoding to obtain the proper value for the classifier.

Pool initialization

The initialization step generates the first pool. The first individual generated (I_{11}) is created with default values of hyperparameters: $g_{11} = (\text{Gini}, \infty, 2, \infty, 0.5)$. The purity of the node is measured with the *Gini* index; the tree can be widened and deepened as needed since the limits for the depth and number of leaves within a node is not fixed and the lowest minimum of samples to split is used; both positive and negative class have the same weight. After training the first tree with these hyperparameters, the remaining individuals are generated considering the actual values of depth and leaves of that first tree as limit. The second individual will be generated with entropy criterion and those limits, while the rest of individuals are generated with random hyperparameters within the limits fixed by the first individual.

For a better understanding of the previous paragraph, we propose a practical case. Given the first individual of the first generation of the meta-learning (I_{11}), the first tree is trained with the specific values of the hyperparameters (*Gini*, ∞ , 2, ∞ , 0.5). Thereafter, the decision tree has a depth of value $\text{depth}(I_{11}) = D$ and a total number of leaves equals to $\text{leaves}(I_{11}) = L$. The second individual (I_{12}) is then trained with the following hyperparameter set: (*entropy*, D , 2, L , 0.5). These limits for the depth and number of leaves of the tree (D and L) will be preserved throughout the process until completion, that is, $I_{ij} = (c, d, s, l, w)$ with $c \sim \{\text{Gini}, \text{entropy}\}$, $d \sim U(1, D)$, $s \sim U(2, \text{training_set_size})$, $l \sim U(1, L)$, and $w \sim U(0, 1)$. In this way, this ad hoc modification will let the meta-learning to better adjust to data set characteristics.

3.3 | Crossover operator

The crossover generates two individuals (I_{kj} and $I_{k,j+1}$) that inherit the hyperparameters given by two parents ($I_{k-1,a}$ and $I_{k-1,b}$), depending on the *crossover probability* (p_c). Concretely, this match is based on a given parameter $u \sim \mathcal{U}(0, 1)$ which follows a uniform distribution. If this value is $u \leq p_c$, the crossover function assigns the same hyperparameter value of the parents to the children. Otherwise, it assigns a linear combination of parents' hyperparameters ($g_{k-1,a}$ and $g_{k-1,b}$), where the parameter $\beta \sim \mathcal{U}(0, 1)$:

$$\begin{aligned} g_{kj} &= \frac{g_{k-1,a} + g_{k-1,b}}{2} + \beta \frac{|g_{k-1,a} - g_{k-1,b}|}{2} \\ g_{k,j+1} &= \frac{g_{k-1,a} + g_{k-1,b}}{2} - \beta \frac{|g_{k-1,a} - g_{k-1,b}|}{2} \end{aligned}$$

After that, genes of the resulting offspring are rounded off and decoded to obtain the proper values for the hyperparameters. In integer genes, the rounded values replaces the decimal ones to ensure a more effective search space.

3.4 | Mutation operator

The mutation operator changes the real membership function hyperparameter values encoded in the chromosome, according to the *mutation probability* (p_m) per individual. The gene (hyperparameter) to be mutated is randomly selected over the five genes. Then, given $u', u'' \sim \mathcal{U}(0, 1)$, the chromosome is mutated as follows:

$$g_{kj} = \begin{cases} g_{kj} + \delta(g_{kj} - \min(h_i)), & u' < 0.5 \\ g_{kj} + \delta(\max(h_i) - g_{kj}), & u' \geq 0.5 \end{cases}$$

where

$$\delta = \begin{cases} -1 + 2u''^{\frac{1}{\mu+1}}, & u'' \leq 0.5 \\ 1 - 2(1 - u'')^{\frac{1}{\mu+1}}, & u'' > 0.5. \end{cases}$$

3.5 | Multiobjective approach

The multiobjective optimization is based on two objective functions to be minimized: f_1 evaluates the accuracy and f_2 the fairness of the model. Thus, f_1 is focused on improving the prediction performance while f_2 is used to mitigate the discrimination of the ML algorithm.

Both concepts of accuracy and fairness can be defined in several ways, referring to different meanings. Although the proposed methodology is totally flexible for using any definition, in this study we focus on two of them. We define y as the binary class label vector where 1 is the positive outcome and 0 is the negative outcome; $\hat{y}$ is the predicted outcome of the ML classifier; z is the associated protected feature of each individual, where 1 is the privileged class.

3.5.1 | Error

We consider the *Geometric Mean* (G-mean) to evaluate the performance of the assessment task. G-mean is also widely used for quantifying the classifier performance in class imbalanced problems, since it evaluates both positive and negative classes. It combines true-positive rate (TPR) ($Pr(\hat{y} = 1|y = 1)$) and true-negative rate (TNR) ($Pr(\hat{y} = 0|y = 0)$):

$$\text{G-mean}(\hat{y}, y) = \sqrt{P(\hat{y} = 1|y = 1) \cdot P(\hat{y} = 0|y = 0)}.$$

By maximizing this measure, we ensure the cost of false-positive and false-negative to be low. Since our method is designed for a minimization problem, we consider the first objective function as the G-mean error, that is, $f_1(\hat{y}, y) = 1 - \text{G-mean}(\hat{y}, y)$.

3.5.2 | (Un)fairness

There is no unified statistical or computational formalization of (un)fairness and thus multiple definitions have been proposed in the late years.²⁵ Indeed, the general consensus is that the meaning and implication of each approach highly depends on the context and consequential decisions associated with an intelligent system.²⁶ Fairness can be procedural or substantive, what is also referred to as equal treatment or opportunity-based versus outcome-based notions of bias.¹⁵ Proposals can be widely categorized as individual, similar individuals will get similar predictions, and group fairness, as equal impact on groups. In real conditions, individual and group fairness are often incompatible objectives^{11,27} and thus the selection criteria corresponds to each particular context.²⁶ For instance, it has been argued that equal opportunity is a suitable metric for designing nondiscriminatory loan strategies²⁸ whereas disparate false positive rate is a widely used metric to quantify discriminator behavior of recidivism algorithms.⁴

For the purpose of this study, we selected one unfairness metric as one of the objective functions for all the data sets but any of the metrics available in the literature could be used (notice that the proposed method optimizes hyperparameters of learners and, therefore, the fairness criteria do not need to be differentiable, thus allowing a wider bank of definitions). We consider the difference of the unfairness measure proposed for avoiding *disparate mistreatment*, defined as false-positive rate (FPR).^{8,27} This definition ensures that misclassification rates are balanced across groups of the protected attribute z :

$$f_2(\hat{y}, y) = \text{FPR}_{\text{diff}}(\hat{y}, y) = |P(\hat{y} \neq y|z = 0, y = 0) - P(\hat{y} \neq y|z = 1, y = 0)|.$$

3.5.3 | Domination criterion

In case of using logistic regression as base learner, the domination criterion is standard, that is, a set of hyperparameters X dominates other set Y if the classification model generated from X is better or equal than the one generated by Y in both accuracy and fairness and strictly better in at least one of them.

In case of using decision trees, the domination criterion is more sophisticated to achieve a more effective optimization. Given X the genotype (learner's hyperparameters) and Y the

phenotype (classification model, i.e., decision tree), the $f: X \rightarrow Y$ map obtained by the proposed method is characterized by being a noninjective nonsurjective function. It is not injective as different values of hyperparameters can lead to obtain exactly the same decision tree. It is not surjective as the image (set of all possible decision trees generated by our method) does not fill the whole codomain, that is, it is not possible to obtain any decision tree, only those generated by the learner. The cardinality of Y is much more lesser than the cardinality of X .

As a result, there are many different individuals that generate exactly the same decision tree, and so the same objective functions. This impairs the search process as variations generated by crossover and mutation do not change the objective functions. To palliate this effect, we have improved the domination criterion as follows. Once two individuals have the same values for both objectives, we consider that the individual that generates the tree with the lowest number of leaves dominates the other one. In case of a tie also in this value, the individual with the lowest value of the hyperparameter `max_leaf_nodes` is considered to dominate the other one.

4 | EXPERIMENTAL ANALYSIS

In this section we first describe the data sets used for assessing the proposed methodology. After that, we define the parameter setup used in these experiments. Finally, the obtained results and its analysis is provided.

4.1 | Data sets

We ran experiments based on five realworld data sets from different domains like salaries, recruitment processes, credit risks, or recidivism risk assessment. These data sets have been widely used as benchmarking in state-of-the-art in fairness.¹⁶ They are freely available in a Github repository¹. A brief description of the data set context is given below:

- *Adults*: This data set contains demographic information about the U.S. citizens in 1994². There are 32,561 instances and 14 attributes. The prediction task is to asses whether an individual earns more (positive class) or less (negative class) than \$50K per year. The protected attribute considered is *race*.
- *German*: It contains financial information about individuals³. There are 1000 instances and 20 attributes. The prediction task is to assess the credit risk of individuals. The protected attribute considered is *age*.
- *ProPublica*: This data set is about the performance of COMPAS algorithm, a statistical method for assigning risk scores within the U.S. criminal justice system created by Northpointe. It was published by ProPublica in 2016,⁴ claiming that this risk tool was biased against African-American individuals. In this data set, they analyzed the COMPAS scores for “risk of recidivism” and checked to see how many were charged with new crimes over the

¹<https://github.com/algofairness/fairness-comparison/tree/master/fairness/data>. Last date accessed: November 8, 2020

²<http://archive.ics.uci.edu/ml/datasets/adult>

³[http://archive.ics.uci.edu/ml/datasets/statlog+\(german+credit+data\)](http://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data))

next 2 years. It contains individuals from the Broward County (Florida) in 2013 and 2014. There are 7214 individuals containing 52 attributes. From these attributes, we have used the following 12 in the experiments of this paper¹⁶: *sex*, *age*, *age_cat*, *race*, *juv_fel_count*, *juv_misd_count*, *juv_other_count*, *priors_count*, *c_charge_degree*, *c_charge_desc*, *decile_score*, *score_text*. The prediction variable is whether the individual will be rearrested in 2 years or not. The protected attribute is *race*.

- *ProPublica violent*: This data set describes the same scenario as the previous one, but in this case the outcome is whether the rearrest happened within 2 years was for a violent crime.⁴ It contains 4743 individuals and also the 12 attributes. The protected attribute is also *race*.
- *Ricci*: This data set comes from labor law case from the United States, where several fire-fighters from New Haven (Connecticut) claimed for disparate impact on the promotion decision. It contains the scores obtained in the exam taken to be promoted.³ There are a total number of 118 rows and 4 attributes. The protected attribute is *race*.

Each data set is preprocessed to assure that the input data satisfies the classifier requirements by removing features that should not be used for the classification task, imputing missing values or transforming features like dates, and so forth. We also transform all the protected attributes into binary (e.g., “white”–“not white,” “younger than 25 years old”–“older than 25 years old,” “caucasian”–“not caucasian”). Table 1 shows the number of features selected for each data set and class distribution.

4.2 | Parameter setup

The experiments are replicated 10 times with different seeds to ensure stability and reproducibility. In each seed, the training (75%) and testing sets (25%) are randomly sampled. Then, the training set is split again in the learning (75%) and validation (25%) sets. Therefore, the testing set is never used in the learning phase. The parameters for the evolutionary method are set as follows:

- 300 generations ($G = 300$),
- 50 individuals ($N = 50$),
- 1 as crossover probability ($p_c = 1$),
- 0.3 as mutation probability ($p_m = 0.3$),
- 5 as mutation parameter ($\mu = 5$).

The code is implemented in Python using libraries such as pandas for data processing, `sklearn.linear_model.LogisticRegression` for logistic regression classifier, `sklearn.tree.DecisionTreeClassifier` for decision tree classifier (CART algorithm) and numpy for numerical processing. The original code of the NSGA-II algorithm is available at github.com/baopng/NSGA-II (last date accessed: June 9, 2020). This study complies with research reproducibility principles. Code and data are made open and available in a public repository: https://github.com/anavaldi/fairness_nsga (last date accessed: November 8, 2020).

TABLE 1 Summary of data sets

Data set	# Features	# Positive	# Negative
Adults	14	7841	24,720
German	20	300	700
ProPublica	12	3251	3963
ProPublica Violent	12	775	3968
Ricci	4	56	62

4.3 | Analysis of results

In this section, we empirically study the limits of the accuracy-fairness tradeoff. We first analyze the properties of the Pareto optimal solutions obtained when optimizing both together. We also analyze the relationship between decision tree learner's hyperparameters and measures' values. Finally, we present the convergence properties of the meta-learning approach.

4.3.1 | Analysis of accuracy-fairness tradeoff

The averaged results over 10 runs are shown in Tables 2 and 3 for the five real-world problems. To represent the average distribution of the obtained results, we have computed the average of the 10 runs at minimum value of error in validation data set (Error_v), 25th percentile (Q_1), 50th (Q_2), 75th (Q_3) and maximum value of error. As the set of inferred solutions are Pareto efficient, the corresponding values of unfairness are reversely sorted. In the case of the two ProPublica problems, the results obtained by COMPAS are also included to better understand the room for improvement in those cases.

The obtained results in Ricci are very particular. We found that this problem is very easy to be solved in terms of accuracy, that is, it is possible to obtain solutions with almost zero error and, therefore, almost one unfairness. In fact, in some partitions the solution found was perfect. Consequently, the multiobjective optimization tends to obtain very spread Pareto solutions, so we decided to leave this problem out of the rest of the analysis.

While the validation data set is used to guide the meta-learning algorithm, the test data set is never used. When comparing validation and test columns, we observe that, although the scores in test are slightly worse than validation (as expected), the Pareto efficiency in test also remains in both learners (logistic regression and decision trees), which shows the robustness of our methodology. Yet the results are overfitted regarding the unfairness measure (i.e., strong differences between Unfairness_v and Unfairness_t) when it comes to very low values, being this effect more pronounced in the decision tree case.

When comparing the average results of the first ($\min$) and 50th (Q_2) positions of error (Q_1 in Adult), we are able to estimate the percentage of accuracy that needs to be killed to improve fairness. In the case of logistic regression, the accuracy lost in test (Error_t) is 17%, 18%, 27%, and 28% in Adult, German, ProPublica, and ProPublica violent problems, respectively, while the fairness improvement in test is 71%, 41%, 16%, and 29%, respectively. In the case of decision trees, the accuracy lost in test is 6%, 10%, 8%, and 13% in Adult, German, ProPublica, and ProPublica violent problems, respectively, while the fairness improvement is of 81%, 66%, 54%,

TABLE 2 Accuracy-fairness tradeoff (*how fair can we go*) in each real-world problem with logistic regression classifiers

	Error _v	Unfairness _v	Error _t	Unfairness _t
Adult				
min	0.22171	0.14605	0.22833	0.23870
Q ₁ (25%)	0.24668	0.08250	0.26651	0.16854
Q ₂ (50%)	0.30253	0.04525	0.33614	0.06859
Q ₃ (75%)	0.38124	0.01792	0.44643	0.01827
max	0.55002	0.00000	0.66207	0.00051
German				
min	0.25026	0.25581	0.29674	0.40670
Q ₁ (25%)	0.26723	0.19179	0.31927	0.22520
Q ₂ (50%)	0.29794	0.13682	0.35017	0.16746
Q ₃ (75%)	0.38014	0.08188	0.40965	0.10518
max	0.59302	0.02313	0.55505	0.04560
ProPublica				
min	0.31820	0.10792	0.33085	0.38621
Q ₁ (25%)	0.34400	0.06296	0.35429	0.09125
Q ₂ (50%)	0.42376	0.04289	0.41944	0.06044
Q ₃ (75%)	0.55705	0.02613	0.53796	0.04111
max	0.78094	0.00343	0.80841	0.00843
COMPAS	0.35002	0.12519	0.34759	0.14751
ProPublica violent				
min	0.30517	0.11799	0.33315	0.20196
Q ₁ (25%)	0.34591	0.06655	0.38866	0.07698
Q ₂ (50%)	0.40163	0.03914	0.42673	0.05837
Q ₃ (75%)	0.49521	0.01753	0.52636	0.02962
max	0.70346	0.00061	0.71792	0.00727
COMPAS	0.32388	0.13474	0.33494	0.13897
Ricci				
min	0.00000	1.0000	0.09053	0.83833
Q ₁ (25%)	0.02715	0.73786	0.10505	0.78631
Q ₂ (50%)	0.04870	0.57073	0.12571	0.73429
Q ₃ (75%)	0.06685	0.50466	0.14074	0.71394
max	0.08776	0.43860	0.14821	0.69359

Note: The table shows the averaged distribution of error (I–G mean) and unfairness (FPR_{diff}) measured in the Pareto optimal solutions for validation (*v*) and test (*t*).

and 53%, respectively. This reveals how the meta-learning algorithm is able to balance accuracy and fairness in practice. It is clear how decision trees are able to get better fairness levels with a moderate degradation of accuracy. Indeed, logistic regression has difficulties to improve the fairness without noticeably degrading the accuracy in the two ProPublica datasets. This gives us an idea of how it is possible to optimize the ML process to generate fairer solutions, specially

TABLE 3 Accuracy-fairness tradeoff (*how fair can we go*) in each real-world problem with decision tree classifiers

	Error _v	Unfairness _v	Error _t	Unfairness _t	Depth	Leaves
Adult						
min	0.17644	0.06743	0.18238	0.07218	8.4	95.5
Q ₁ (25%)	0.19374	0.04036	0.19412	0.05822	12.1	211.2
Q ₂ (50%)	0.21715	0.02423	0.22220	0.04577	14.5	352.4
Q ₃ (75%)	0.26488	0.00971	0.26804	0.02620	16.8	518.9
max	0.35766	0.00034	0.35759	0.00794	22.6	945.6
German						
min	0.26780	0.12406	0.32393	0.16990	6.9	22.3
Q ₁ (25%)	0.27830	0.08135	0.34387	0.13916	7.9	28.1
Q ₂ (50%)	0.29442	0.04411	0.35488	0.11279	9.1	34.0
Q ₃ (75%)	0.31977	0.01989	0.37343	0.07821	9.4	40.2
max	0.38101	0.00099	0.43214	0.02597	10.3	47.6
ProPublica						
min	0.32759	0.12471	0.33676	0.12871	6.7	50.5
Q ₁ (25%)	0.34078	0.08052	0.35094	0.08936	10.0	145.2
Q ₂ (50%)	0.35572	0.03476	0.36223	0.07011	12.1	238.4
Q ₃ (75%)	0.38492	0.01362	0.39121	0.04591	14.4	312.0
max	0.39997	0.00293	0.40881	0.03026	16.7	467.4
COMPAS	0.35002	0.12519	0.34759	0.14751	–	–
ProPublica violent						
min	0.31366	0.10367	0.33176	0.10261	6.2	34.6
Q ₁ (25%)	0.33651	0.06047	0.35422	0.07879	8.9	71.7
Q ₂ (50%)	0.35388	0.03446	0.37430	0.05461	10.8	109.6
Q ₃ (75%)	0.38638	0.01011	0.41021	0.03251	12.2	148.2
max	0.48942	0.00021	0.50264	0.01794	14.4	210.1
COMPAS	0.32388	0.13474	0.33494	0.13897	–	–
Ricci						
min	0.04487	1.0000	0.12249	0.80222	1.8	2.9
Q ₁ (25%)	0.09006	0.71526	0.15782	0.66326	2.1	3.4
Q ₂ (50%)	0.13134	0.46007	0.18936	0.54931	2.4	3.8
Q ₃ (75%)	0.17195	0.30838	0.21820	0.43881	2.6	4.1
max	0.21268	0.15669	0.24781	0.32831	2.9	4.4

Note: The averaged distribution of error (I-G-mean) and unfairness (FPR_{diff}) measures in the obtained Pareto optimal solutions for validation (v) and test (t) data sets are shown. Depth and leaves (complexity) are the actual values of the generated decision trees.

when using decision trees, without an excessive loss of precision, which should encourage ML designers to incorporate fairness criteria into these processes.

Focusing on the two ProPublica problems, where the prediction made by COMPAS is widely known, we can analyze the accuracy and fairness achieved by the Northpointe's software when assessing a criminal defendant's likelihood to re-offend. We can observe that, with a similar accuracy, the fairness of the solutions got by our methodology is much fairer than the

obtained by COMPAS regardless of the classifier used. This demonstrates the improvement margin of fairness in these problems when guiding the ML process by unbiased measurements. When using decision trees, if we interpolate the fairness scores got by our methodology for an accuracy equal to COMPAS's, the test results would be $(\text{Error}_t, \text{Unfairness}_t) = (0.3476, 0.0987)$ in ProPublica and $(\text{Error}_t, \text{Unfairness}_t) = (0.3349, 0.0992)$ in ProPublica violent, showing that our method improves the fairness of COMPAS's solutions in 67% and 71%, respectively, without compromising accuracy.

When analyzing the performance of solutions, we are additionally concerned with transparency of the classifiers. Indeed, in the problems considered in our experimental analysis, where wrong outcomes may discriminate disfavored social groups, to understand the reasoning behind a machine decision is critical. Therefore, we analyze in which degree the Pareto optimal models are also easy to interpret in the case of considering decision trees as classifiers. The fact of being using this kind of structure to represent the knowledge helps to understand the machine decision criteria compared with other black-box models, but the complexity of these trees will also influence on its interpretability, as an excessively fine-grain decision boundary (high number of leaves) and complex multivariate conditions (high depth of the tree) would be hardly understandable.

Analyzing the complexity results in Table 3, we observe that the number of leaves is relatively low in the most accurate solutions, but tends to increase as fairness improves. This effect shows that the method needs to use more leaves to improve fairness with a minimum loss of accuracy. This is an expected result since equalizing FPR between the two people groups forces a finer partitioning of data. The high depth with a relatively low number of leaves suggests the construction of unbalanced decision trees (keep in mind that a perfectly balanced binary tree would need 2^{depth} leaves, which is very far from what we get). That is, some few leaves need a high depth (i.e., extensive multiple conditions) to be effective.

Analogously, it is well known that a lower error implies a higher complexity, so it is curious to observe that this relation is not shown in the obtained results. The reason is simply that the complexity (number of leaves and depth) is not considered as a criteria to be optimized by our methodology, so this variable is freely adapted to the two contradictory objectives (accuracy and fairness), both of them demanding higher complexity to be reached. It seems that the fairness objective ends up winning the battle. In other words, the algorithm finds it harder to improve fairness than accuracy with a reduced complexity. Nevertheless, this interesting effect deserves a deeper study that would divert us from the main goal of our research in this paper, so we leave it as a further research line.

To better understand the behavior of the proposed method, Figures 2 and 3 plot the obtained Pareto optimal solutions with the two base classifiers considered in this paper, with gray dots being the solutions of each run and brown (logistic regression) or violet (decision trees) dots connected by lines represent the average Pareto front. This average Pareto is obtained by first getting the rounded mean number of different solutions n (which corresponds to the number of solid dots) and then obtaining the average values at n different percentiles positions equally distributed. For example, if we have three runs where we got 3, 5, and 7 Pareto optimal solutions, we would obtain the $n = 5$ evenly distributed percentiles (i.e., 1th, 25th, 50th, 75th, and 100th) with linear interpolation between adjacent ranks in each run and then calculate the average value for each percentile. The interquartile range ($Q_3 - Q_1$) of the error is represented with the light brown/violet area in the figures.

The spread of the dots (especially in the fairness dimension) and the width of the interquartile range suggest us that the attainable levels of accuracy and fairness is quite sensitive to

the data set partitions into training and test. This is particularly serious in German. The exception is represented by Adult, where the solutions in different data partitions are very compact. This may be due the fact that Adult has a considerably high number of data so that the bias of the data partitioning is mitigated. As our methodology splits the training data into learning and validation, it suffers when very little data is available, as in German.

As we can observe from the plotted Pareto fronts, the contradictory condition between accuracy and fairness is clear: more accuracy implies less fairness, and vice versa, as analyzed in previous works.^{19,20} Although what is really interesting to analyze is the shapes of the averaged Pareto fronts as they provide valuable information about how the combination data set and classifier is working. In fact, beyond generating a wide repertoire of solutions with different balances of accuracy and fairness, our methodology also returns a greater understanding of the problem by explaining how these contradictory criteria are related.

Let us take as example the ProPublica problem with decision trees (Figure 3C). The accuracy–fairness relation is rather linear in the range [0.026, 0.125] of unfairness, that is, range [0.328, 0.361] of error. Then, we see a clear knee of the curve below an unfairness of 0.026, meaning beyond this threshold, improving a bit the fairness has a relatively high cost in accuracy. Similar conclusion can be taken in the other problems, where the unfairness threshold is around 0.01 in Adult, and 0.02 in German and ProPublica Violent with decision trees. In the case of logistic regression, the knees are around 0.03 in the two ProPublica and 0.12 in German. This knowledge could be used by other researchers and practitioners to set different fairness requirements depending on the problem and the type of classifier that is being used.

Finally, for a better comparison of the behavior of the two base classifiers analyzed in this paper, Figure 4 shows the average Pareto obtained in each problem with logistic regression and decision trees. As we already mentioned when analyzing the tables of results, we can observe that the use of decision trees as base learners makes our proposed methodology to perform better. This is very clear in adult data set, where the decision trees' Pareto completely dominates the logistic regression's one. In the other three cases, however, we can see how logistic regression can achieve a slightly better accuracy (with subsequent worse fairness). Nevertheless, when it comes to fairness, decision trees allow a greater improvement with a more restrained degradation of accuracy than logistic regression. We believe this effect is related with the highest capability of decision trees to partition the attribute space, as they are able to fix decision boundaries that compartmentalize data with a finer grain than the linear planes fixed with logistic regression. This finer division of data allows to distribute better the FPR between groups, thus being able to better reduce disparate mistreatment.

4.3.2 | Analysis of learner's hyperparameters

As we are proposing a meta-learning method that indirectly controls the generated classification models by tuning the hyperparameters of the learner, we are also interested in assessing the impact of learner's hyperparameters on the performance. We will focus this study on decision trees. We have already discussed in the previous section the effect of demanding optimal fairness in the complexity of the trees (good fairness needs higher number of leaves). Here we analyze the effect of two other hyperparameters: `min_samples_split` and `class_weight`. We did not find significant results in the fifth hyperparameter

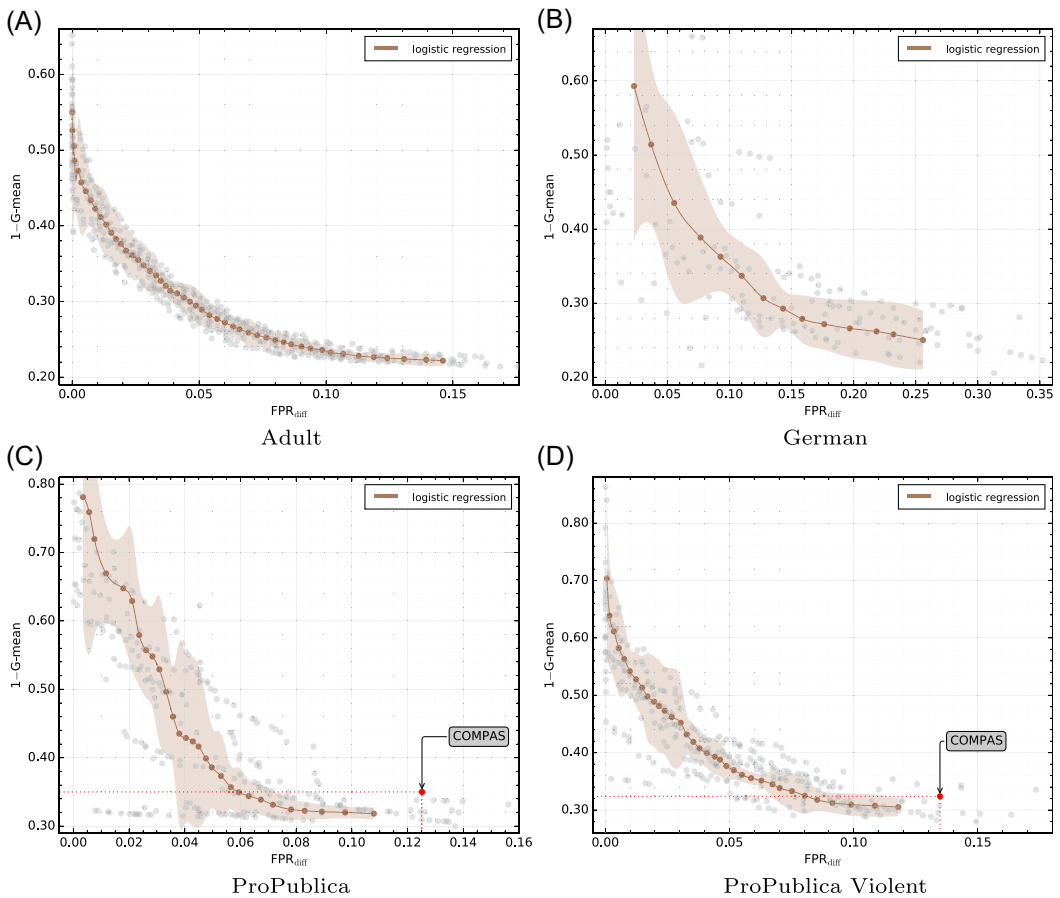


FIGURE 2 Solutions obtained with logistic regression classifiers. Gray dots represent Pareto optimal solutions—minimizing error (I-G-mean) versus unfairness (FPR_{diff})—found by the proposed algorithm in different problems. Brown dots indicate the average Pareto set, which is a way of representing *how fair can we go* with logistic regression in a specific problem or, in other words, which shape takes the accuracy-fairness tradeoff with such a kind of classifier. Light brown area is the interquartile range. Our methodology is effective to find a wide spread of solutions that are accurate and fair at the same time. In the two ProPublica datasets, the meta-learning algorithm also finds better solutions than the obtained by COMPAS (red dots), showing that there is a wide range of possibilities to be fairer without worsening accuracy [Color figure can be viewed at wileyonlinelibrary.com]

(criterion). Figure 5 shows the values of these two hyperparameters in the obtained Pareto optimal solutions of ProPublica violent. The mean values over all the runs is plotted as lines and dots, while the shaded areas and error bars correspond to the *SD*. Blue color is used for error and red color for unfairness, both in test data sets ($Error_t$ and $Unfairness_t$).

In the case of `min_samples_split`, the results confirm our guess that to improve fairness it is necessary to deepen certain branches of the tree, so that a low value of the limit of samples needed to divide a node helps to generate fairer trees. It is interesting to see here how a high value of this limit hurts fairness a lot but does not influence accuracy.

With regard to `class_weight`, which controls the importance of the positive class (and reversely the negative one), the effect is as follows. In accuracy, a higher weight of the positive class implies generating more accurate solutions in this imbalance data set (there are five times

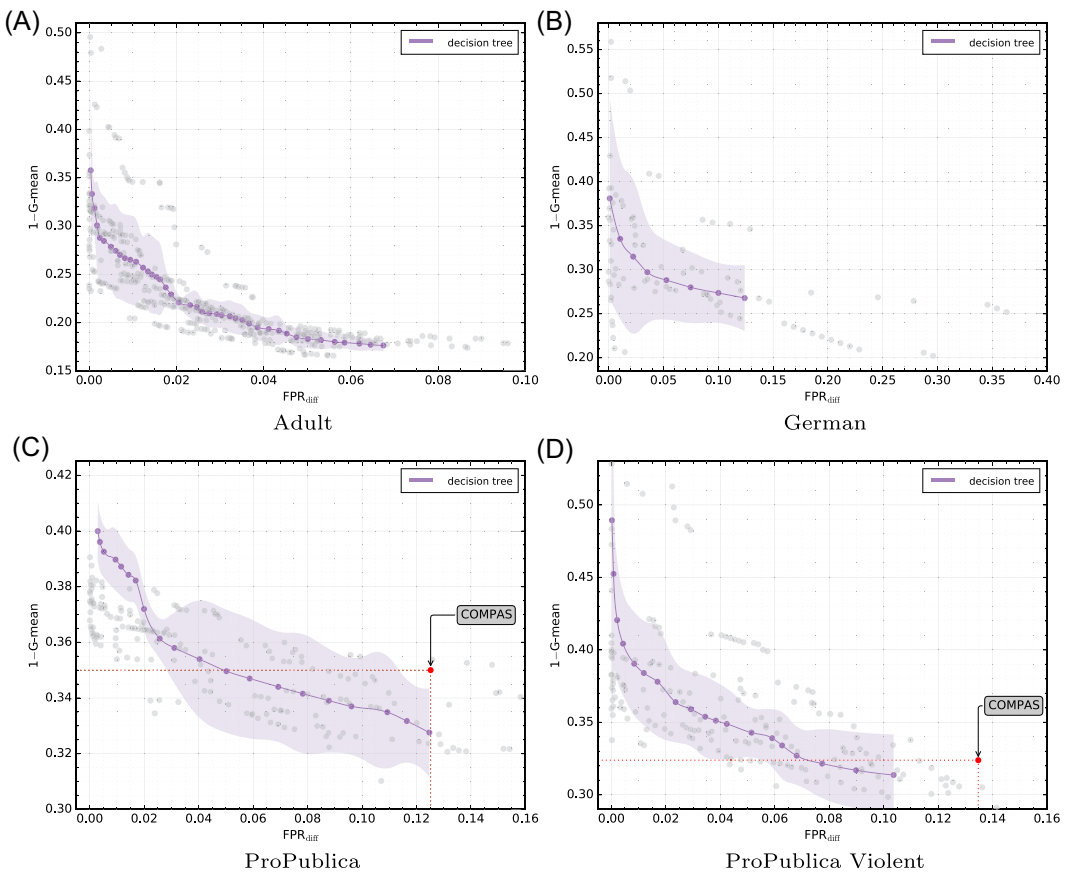


FIGURE 3 Solutions obtained with decision tree classifiers. Gray dots represent Pareto optimal solutions—minimizing error (1-G-mean) versus unfairness (FPR_{diff})—found by the proposed algorithm in different problems. Violet dots indicate the average Pareto set, which is a way of representing *how fair can we go* with decision trees in a specific problem or, in other words, which shape takes the accuracy-fairness tradeoff with such a kind of classifier. Light violet area is the interquartile range. Our methodology is effective to find a wide spread of solutions that are accurate and fair at the same time. In the two ProPublica data sets, the meta-learning algorithm also finds better solutions than the obtained by COMPAS (red dots), showing that there is a wide range of possibilities to be fairer without worsening accuracy [Color figure can be viewed at wileyonlinelibrary.com]

more of the negative than the positive). This makes sense as G-mean measure rewards balanced predictive precision in the two classes, so making more important the minority (positive) class helps to this goal. This hyperparameter has the contrary effect in fairness. Here, fairer solutions are obtained when a positive class weight in [3, 5] is given (moreover, with a low variance that ensure statistical significance), that is, to decrease the importance of the positive class (which in the analyzed problem means that the criminal defendant reoffends) reduces the false-positives, which makes easier to generate decision trees with a better balance of FPR between the two groups (Caucasian vs. rest of ethnics). In other words, giving less credibility to the positive class (reoffend) allows for fairer classifiers. However, we cannot ignore that this could also be a side effect of the Pareto efficiency followed by the optimization process.

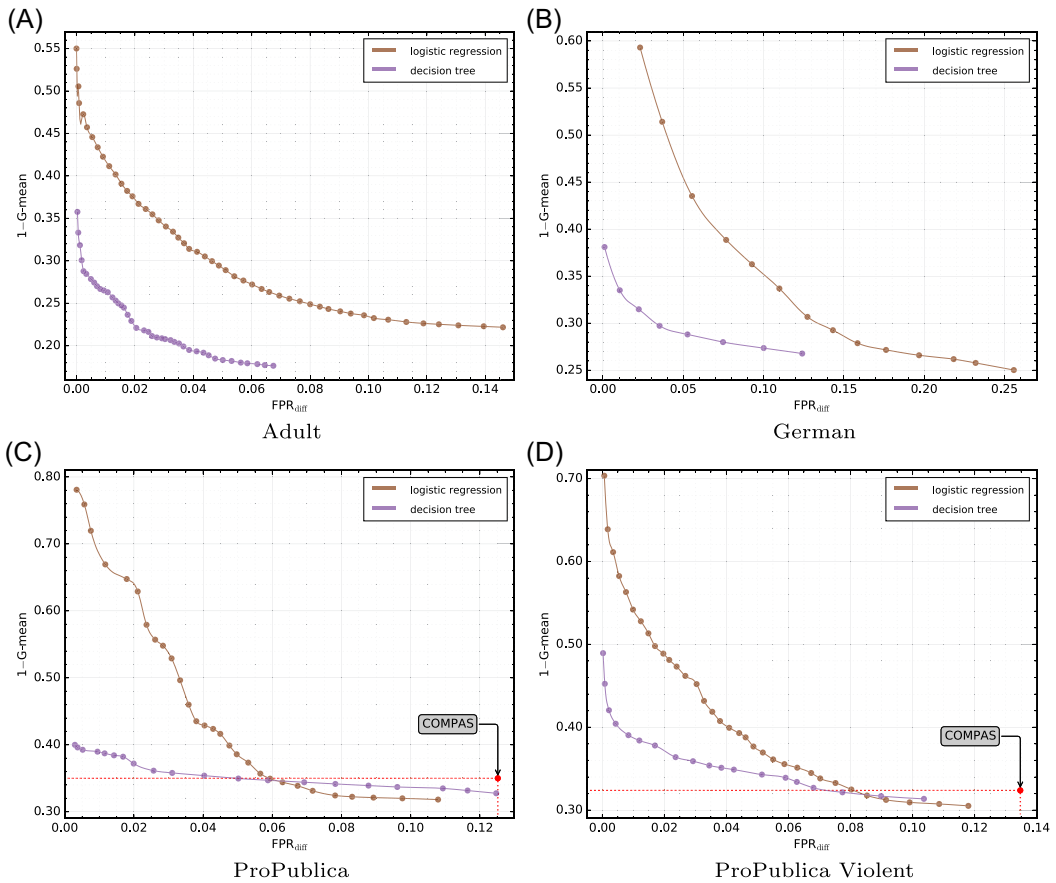


FIGURE 4 Comparison between using logistic regression versus decision trees as classifiers in the proposed meta-learning algorithm. Average Pareto sets are plotted [Color figure can be viewed at wileyonlinelibrary.com]

4.3.3 | Analysis of convergence

An algorithm converges when there is no significant improvement in the values of the objective functions of the population from one to the next generation. This aspect is important to be studied to assess the efficiency of the method. At the same time, its analysis can reveal the resistance of each problem to allow improvements of the accuracy and fairness measures.

In multiobjective optimization, convergence is more complex to analyze as many optimal solutions evolve at the same time. To summarize the behavior of the process, Figure 6 presents the mean, Q_1 and Q_3 of the two objectives (error and unfairness in validation set) for the obtained Pareto set at each generation (averaged results over 10 runs are plotted) when using decision tree as base classifier. In some way, the mean gives an idea about the quality of the solutions (the lower the better) while the interval $[Q_1, Q_3]$ represents the diversity of the Pareto sets (the wider the better). In Ricci, the algorithm fully converges very quickly (these values do not change at all after 27 generations), so we omit this plot for the sake of clarity of the paper.

We observe that low unfairness is faster to get than low error, so in the first third of the evolution good fairness is reached in all the problems, while the accuracy is slowly improved until the end of the process. Adult has the most stable convergence of the four shown problems

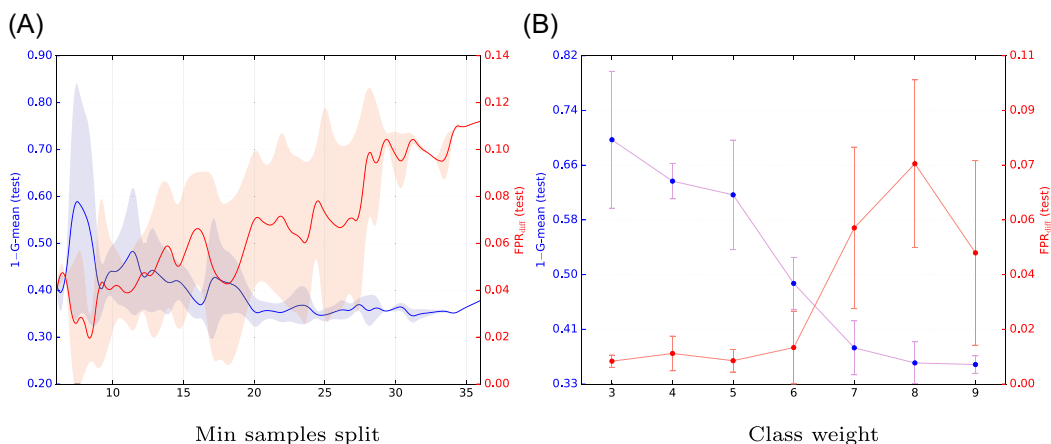


FIGURE 5 Effect of two hyperparameters of the decision tree classifier in ProPublica violent. Notice how different values of them impacts varyingly on accuracy and fairness. Low threshold to split a node and low weight of the positive (minority) class favor the generation of decision trees with a good fairness [Color figure can be viewed at wileyonlinelibrary.com]

due to the reduced bias in data partitions as above said. German also converges very well, but with a slight improvement of accuracy in the last 40 generations at the expense of making fairness slightly worse. ProPublica shows the most continuous convergence where accuracy and fairness are persistently improved. In ProPublica Violent, good fairness is very quickly obtained while accuracy is continually enhanced.

5 | CONCLUSIONS

In this study we propose a meta-learning multi-objective optimization algorithm to explore the boundaries of fairness in real-world problems. We present a methodology that (1) enables standard ML algorithms to be fairness-aware, (2) obtains the experimental frontier of the accuracy-fairness tradeoff, (3) uses interpretable models as base learners to comply with transparency values, and (4) converges rapidly to optimal solutions. To the best of our knowledge, this is the first work that proposes both accuracy and fairness as objective functions for a multiobjective ML approach.

Accuracy versus Fairness: Throughout the experimental analysis, we show the optimal fitness that can be achieved by optimizing the geometric mean of the predictive precision of each class versus FPR equality of the groups, that is, no disparate mistreatment as defined in Reference [21]. The cost in accuracy when satisfying fairness criteria has been theoretically studied (e.g., References [17,21]). These studies demonstrate the existence of an unavoidable tradeoff between accuracy with respect to the target variable and fairness with respect to the sensitive attribute. That is, when one objective is improved by the model the second one is penalized. Or what is the same, these two objectives are contradictory. Based on this assertion, we design in this paper an optimization process able to push both objectives to the frontier where the mentioned Pareto efficiency is reached, thus returning a plethora of solutions with different accuracy-fairness balances. Besides, the experimental analysis shows how fair can we go in a specific problem by logistic regression and decision trees, providing further insight about the capability of standard ML algorithms to get good fairness and the flexibility of the

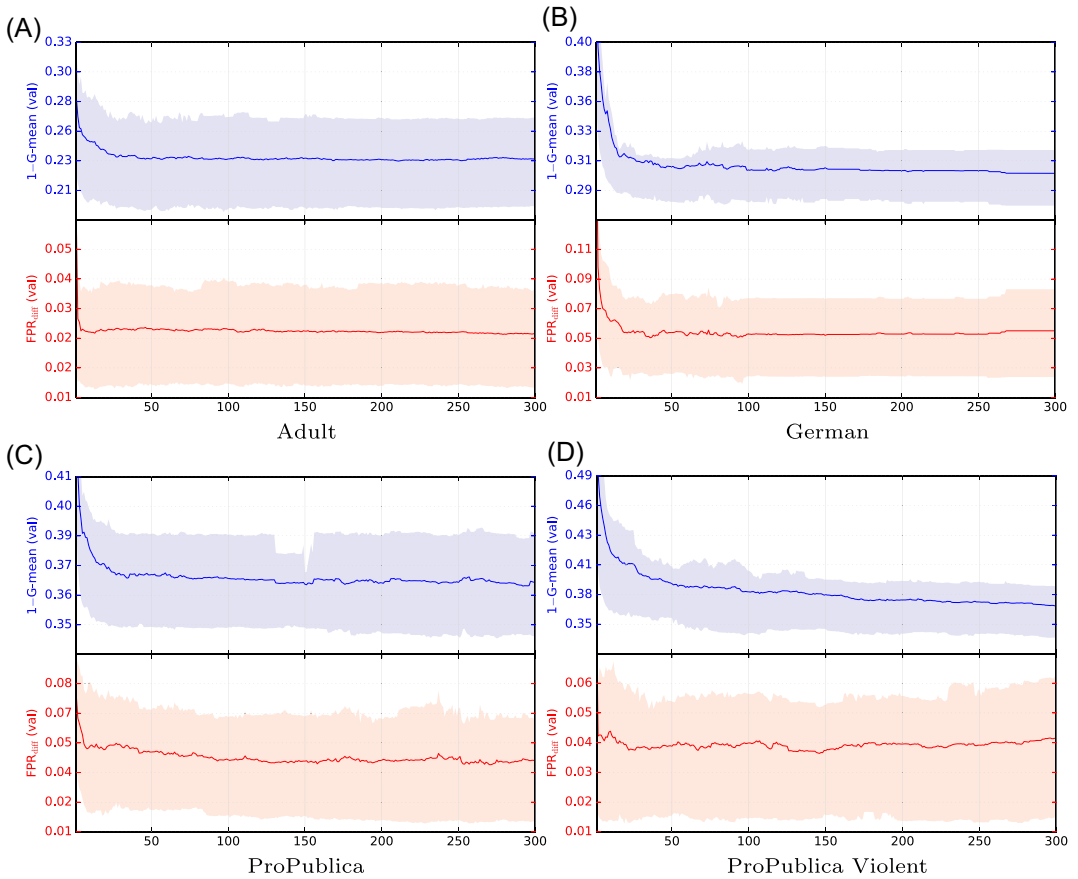


FIGURE 6 Evolution of nondominated solutions through 300 generations of the meta-learning algorithm with decision tree as base classifier. To represent the distribution of these Pareto sets, mean (line) and $Q_1 - Q_3$ (area) of error and unfairness objectives (i.e., in validation set) averaged over 10 runs are plotted [Color figure can be viewed at wileyonlinelibrary.com]

problem (data set) to allow this. In fact, we show how the fairness of the COMPAS solutions in the two ProPublica datasets can be improved by about 70% without compromising accuracy.

Logistic Regression versus Decision Trees: The paper presented two examples of the methodology by using either logistic regression or decision trees as base classifiers within the proposed meta-learning. The comparative experimental analysis yields interesting results. It can be observed that decision trees are superior than logistic regression as far as fairness is concerned; in other words, with decision trees we can obtain fairer solutions with a lower accuracy sacrifice. This effect may be due to the fact that the decision boundaries managed by trees can split data in a finer way, which is a competitive advantage when it comes to fairness, as in this way it is possible to better distribute data to balance the FPR between groups, thus favoring a fairer treatment. However, it is important to emphasize that these results are achieved thanks to the intensive tuning of hyperparameters that our methodology performs, so a standard application of these two learners with default hyperparameters values might not make the remarkable difference obtained in this paper. What is clear is that by learning decision trees there is much more room for improvement of the fairness with a restrained loss of accuracy.

Fairness versus Transparency: As it is well known in ML, decision trees can improve accuracy (at least, while the sweet spot without overfitting is reached) often by increasing the model complexity (i.e., tree depth and number of leaves). Moreover, we believe that, to improve fairness, the decision tree needs to be deeper for a fine-grain data partition to hold misclassification parity between different groups having different values of the sensitive attribute. Therefore, both accuracy and fairness demand more complex decision trees. When optimizing accuracy and fairness together, we find that the process tends to solve the conflict by generating more complex trees in fairer solutions even when their accuracies are not so good. This may be due to the fact that, when optimizing learner's hyperparameters as our methodology do, fairness is mainly reached by more complex trees while there are other chances of improving accuracy by fine-tuning the remaining hyperparameters.

Convergence: Evolutionary algorithms are sharply criticized because of its low convergence in many problems. Nevertheless, we show that this methodology early achieves optimal solutions. Objective functions cooperate to generate good solutions in the first generations, but they compete to obtain optimal solutions at the end.

Future Work: Although we know that technology interventions alone will not address social injustice, there are several interesting directions highlighted by our findings. From the obtained results, it is clear further research is needed to understand the role of transparency (in terms of model complexity) in the accuracy-fairness tradeoff. Therefore, we propose to add the complexity of the trees as a third objective function (f_3). Regarding the fact that fairness can be defined in multiple ways, we plan to develop further analysis with different measures of mistreatment. In relation to claims by Reference [17], it would be interesting to study data set properties, such as correlation of the sensitive attribute with the target variable. We are aware that the experiments presented in this study only include one binary sensitive attribute. We propose to consider more attributes in further experiments to analyze how convergence is affected. Differential fairness²⁹ is a growing concept highly related with this study, which addresses intersectionality. We propose to run new experiments of our meta-learning algorithm proposing this new fairness definition. Finally, it is worth mentioning that our approach is completely flexible, and its design allows the use of any type of classifier and hyperparameters, which serving as a tool to experimentally analyze several dimensions of the behavior of ML methods.

ACKNOWLEDGMENTS

This article was written in the context of the SECURITY FLOWS project, which received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 819213). The work of Javier Sánchez-Monedero was supported by a Starting Grant from the ERC under grant number 759903. The work of Jorge Casillas was supported by the Spanish Ministry of Science and Innovation, Institute of Health Carlos III, grant no. PI20/01435. Finally, Ana Valdivia would like to acknowledge the Spanish welfare system, which supported the first part of this study while she was unemployed.

REFERENCES

1. O'Neil C. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. New York, NY: Crown Publishing Group; 2016.
2. Eubanks V. *Automating Inequality*. New York, NY: St. Martin's Press; 2018.
3. S. C. of the United States, Ricci v. DeStefano. 557 U.S. 557 (2009)174.

4. Angwin J, Larson J, Mattu S, Kirchner L. Machine bias. ProPublica. May 23, 2016.
5. Bolukbasi T, Chang K-W, Zou JY, Saligrama V, Kalai AT. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In: *30th Conference on Neural Information Processing Systems (NIPS 2016)*. Barcelona, Spain; 2016:9.
6. Kearns M, Neel S, Roth A, Wu ZS. Preventing fairness gerrymandering: auditing and learning for subgroup fairness. In: Dy J, Krause A, eds. *Proceedings of the 35th International Conference on Machine Learning, Vol. 80 of Proceedings of Machine Learning Research*, Stockholm Sweden: PMLR, Stockholmssmässan; 2018: 2564-2572.
7. Buolamwini J, Gebru T. Gender shades: intersectional accuracy disparities in commercial gender classification. In: *Conference on Fairness, Accountability and Transparency*; 2018:77-91.
8. Zafar MB, Valera I, GomezRodríguez M, Gummadi KP. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In: *Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee*; 2017: 1171-1180.
9. Xia B, Yin J, Xu J, Li Y. WE-Rec: a fairness-aware reciprocal recommendation based on Walrasian equilibrium. *Knowl-Based Syst.* 2019; 182:104857.
10. Zehlike M, Hacker P, Wiedemann E. Matching code and law: achieving algorithmic fairness with optimal transport. *Data Mining Knowl Discov.* 2020; 34(1):163-200.
11. Lipton Z, McAuley J, Chouldechova A. Does mitigating ML's impact disparity require treatment disparity? In: *Advances in Neural Information Processing Systems*; 2018:8125-8135.
12. Binns R, Van Kleek M, Veale M, Lyngs U, Zhao J, Shadbolt N. 'It's reducing a human being to a percentage' perceptions of justice in algorithmic decisions. In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*; 2018:1-14.
13. Mitchell M, Wu S, Zaldívar A, et al. Model cards for model reporting. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19*. New York, NY: ACM; 2019:220-229.
14. Selbst AD, Boyd D, Friedler SA, Venkatasubramanian S, Vertesi J. Fairness and abstraction in socio-technical systems. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19*. New York, NY: ACM; 2019:59-68.
15. Sánchez-Monedero J, Dencik L, Edwards L. What does it mean to 'solve' the problem of discrimination in hiring? Social, technical and legal perspectives from the UK on automated hiring systems. In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, FAT* '20*. New York, NY: ACM; 2020: 458-468.
16. Friedler SA, Scheidegger C, Venkatasubramanian S, Choudhary S, Hamilton EP, Roth D. A comparative study of fairness-enhancing interventions in machine learning. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19*, New York, NY: ACM; 2019:329-338.
17. Menon AK, Williamson RC. The cost of fairness in binary classification. In: Friedler SA, Wilson C, eds. *Proceedings of the 1st Conference on Fairness, Accountability and Transparency, Vol. 81 of Proceedings of Machine Learning Research*. New York, NY: PMLR; 2018:107-118.
18. Kamiran F, Calders T, Pechenizkiy M. Discrimination aware decision tree learning. In: *2010 IEEE International Conference on Data Mining*; 2010:869-874.
19. Agarwal A, Beygelzimer A, Dudik M, Langford J, Wallach H. A reductions approach to fair classification. In: Dy J, Krause A, eds. *Proceedings of the 35th International Conference on Machine Learning, Vol. 80 of Proceedings of Machine Learning Research*. Stockholmssmässan, Stockholm, Sweden: PMLR; 2018:60-69.
20. Balashankar A, Lees A, Welty C, Subramanian L. Pareto-efficient fairness for skewed subgroup data. In: *International Conference on Machine Learning AI for Social Good Workshop*, Long Beach; 2019:8.
21. Zafar MB, Valera I, Gomez-Rodriguez M, Gummadi KP. Fairness constraints: a flexible approach for fair classification. *J Mach Learn Res.* 2019;20(75):1-42.
22. Hu L, Chen Y. Fair classification and social welfare. In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, FAT* '20*. New York, NY: Association for Computing Machinery; 2020: 535-545.
23. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput.* 2002;6(2):182-197.

24. Rudin C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat Mach Intell.* 2019;1(5):206-215.
25. Ntoutsi E, Fafalios P, Gadiraju U, et al. Bias in data-driven artificial intelligence systems—an introductory survey. *WIREs Data Mining Knowl Disc.* 2020; 10(3):e1356. <https://doi.org/10.1002/widm.1356>
26. Hellman D. Measuring algorithmic fairness. *Virgin Law Rev.* 2020; 106(4):811.
27. Chouldechova A. Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. *Big Data.* 2017; 5(2):153-163.
28. Hardt M, Price E, Srebro N. Equality of opportunity in supervised learning. In: Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, eds. *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc.; 2016: 3315-3323.
29. Foulds JR, Islam R, Keya KN, Pan S. An intersectional definition of fairness. In: *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE; 2020:1918-1921.
30. Srinivasan N, Deb K. Multi-objective function optimisation using non-dominated sorting genetic algorithm. *Evol Comput.* 1994; 2(3):221-248.
31. Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: empirical results. *Evol Comput.* 2000; 8(2):173-195.

How to cite this article: Valdivia A, Sánchez-Monedero J, Casillas J. *How fair can we go in machine learning? Assessing the boundaries of accuracy and fairness.* *Int J Intell Syst.* 2020;1–25. <https://doi.org/10.1002/int.22354>

APPENDIX A: NSGA-II

The nondominated sorting genetic algorithm (NSGA)³⁰ was one of the first EAs developed for multiobjective problem optimization. Yet this approach was criticized due to: (1) the high computational complexity, (2) the lack of elitism, and (3) the low spread of solutions. Then, the NSGA-II²³ was proposed as a modification to address these disadvantages. To solve (1) the authors proposed a *nondominated* sorting procedure where all the individuals are sorted according to the level of nondominance. Issue (2) is mitigated through an *elitism* strategy that stores all nondominated solutions and avoids removing good solutions from the pool. This aspect also enhances the convergence property of EAs.³¹ Finally, they adapted a suitable automatic mechanism based on the *crowding distance* to ensure diversity in a population and then solve (3). This distance function assigns a distance metric to all individuals within a population and then compares whether two solutions are close enough. A solution with a smaller value is more crowded by other solutions, therefore is more likely to not survive in further populations.

This approach starts by creating a random parent population P of size N . The population is evaluated by the objective functions and sorted following the non-dominance criteria 2.1. After that, each solution is ranked where the first level corresponds to the best individuals, the second level is the next-best set of members, and so on. After that, the binary tournament selection, crossover, and mutation operators are used to create an offspring population. These children are also evaluated by the objective functions and combined together with the previous population. All individuals are then ranked and sorted by the nondomination rank and the crowding distance, which is considered the elitist step. The N -best members are then selected to pass to the following generation that will complete the next population of solutions by applying crossover and mutation operators. Finally, the algorithm ends when the last generation is reached.