FAIR AUTOML THROUGH MULTI-OBJECTIVE OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

There has been a recent surge of interest in fairness measurement and bias mitigation in machine learning, given the identification of significant disparities in predictions from models in many domains. In part, this focused interest is due to early failures of simple attempts at achieving "fairness through unawareness" in practice. Non-sensitive data may be hopelessly coupled with the omitted sensitive factors and systemic bias inevitably poisons the data in ways that may not be recoverable as the resulting model seeks to describe the effects found in the data on which it is trained. An effective way of preventing bias is to provide tools to measure it from multiple perspectives and viewpoints, and to incorporate these measures within Automated Machine Learning (AutoML) algorithms in search of accurate and fair models. The emerging realization of the importance of such metrics demands a long-standing missing feature, namely the ability to handle multiple objectives and constraints at all stages of the ML pipeline. In this paper, we introduce a novel AutoML framework that naturally supports multi-objective optimization. It generates higher-dimensional Pareto fronts and permits a single optimization process to efficiently achieve a proper approximation of the global front that depicts the trade-off among multiple model fairness and model accuracy measures. We show that both model training hyperparameters and fairness mitigation hyperparameters must be explored concurrently in order to characterize this trade-off most effectively. Results from experiments on multiple commonly investigated real-world case studies validate the effectiveness of our approach.

1 INTRODUCTION

It is not uncommon for an academic field to outpace legal and ethical precedence and understanding as the unraveling of the implications of a new capability begins. It is often as the excitement over the power and potential of a new breakthrough technology wanes, that society realizes the potential dark side a new tool might have. This is true in physics, in genetics, and more recently in the world of machine learning (ML) based artificial intelligence. The unparalleled success of deep learning has arguably created a renaissance, refueling interest across the entire field of machine learning. However, the same success has cast a shadow over the field of AI, as society realizes in the near future, many important decisions with regard to human safety, health, and finances may one day be made more accurately by powerful black-box entities, entities of which we have little understanding.

Modern ML algorithms are predominantly single-minded, designed to do whatever it takes to achieve high accuracy with regard to a single unconstrained objective. This inability to account for the plethora of real-life constraints that must be satisfied to deploy an ML model is its Achilles' heel. An important concern in this light is how global accuracy reflects on the short- and long-term consequences to minority groups and marginalized communities if we were to permit ML models to play a pivotal role in decision making. Early research (O'Neil, 2016; Hicks, 2017; Noble, 2018; Eubanks, 2018; Barocas et al., 2019) addresses and exposes the glaring weakness of modern ML tools when it comes to addressing the issues of fairness and model bias to privileged subgroups. The sources of model bias are likely to occur at all stages of the machine learning pipeline: from data collection, to preparation, to model training and model calibration (Saleiro et al., 2018; Bellamy et al., 2018; Agarwal et al., 2018b).

It is becoming clear that machine learning models can no longer simply ignore the presence of multiple objectives and constraints. Handling multiple objectives significantly increases complexity, but as Zitzler et al. (2002) showed with the following theorem, finding a single performance measure that adequately represents the multiple dimensions of the solution space simply isn't possible:

Theorem 1. *In general, solution quality for the m-objective optimization problem cannot be reduced to fewer than m performance measures.*

This implies that algorithms such as Exponentiated Gradients (Agarwal et al., 2018b) will by themselves struggle as they seek to find a single merit function (in this case Lagrangian) balancing all objectives. This desirability function definition will merely target a single point on the Pareto-front and may miss providing the user a complete list of compromises among targeted objective functions. Thus the totality of ML tools, from data preparation, to model tuning and training, to post processing calibration must ultimately support the ability to juggle multiple nonlinear competing functions. An important subcategory of ML tools is AutoML. It is now more important than ever for all AutoML to naturally and robustly support multiple black-box objectives and constraints.

Recent research has emphasized the power of a model-free approach (Xiong et al., 2021) in the context of AutoML showing that employing classic statistical analysis on parameter sensitivity and selection results in a simple algorithm that is competitive with Bayesian search. While Bayesian is the standard in many AutoML systems, its practical use in the multi-objective nonlinearly constrained case is not yet well defined. One weakness is that the Bayesian model must converge before it can be of use. In the multi-objective case, the Bayesian approach can become cumbersome as points must be selected to optimize multiple models, one for each objective and constraint. In contrast, model-free approaches have been supporting multiple objectives and constraints for decades (Deb et al., 2000; Griffin et al., 2011).

Few existing approaches have numerically studied the effect of more than two objectives in a single problem. Those that do simplify the problem while focusing on a single desirability function or are satisfied with finding a few (of the infinite) Pareto points. Because the number of optimization problems would grow like "n choose 2" (with n objectives) without general support, it is important to be able to handle all objective functions and constraints simultaneously. This effectively enables many related optimization problems to be solved simultaneously while sharing computing resources.

The goal of this paper is not to attempt to answer the challenging problem of deciding which fairness metrics are warranted. Rather, we seek to enhance the growing toolbox for end users and data scientists who ultimately must juggle high-dimensional Pareto fronts, derived from these metrics, by leveraging a host of algorithms that are predominantly single objective in nature. We provide data scientists a framework that intercepts, at the highest level, all the sensitive key factors that affect accuracy and fairness currently passed off as the "user's responsibility to control and adjust." Our approach optimizes the bias mitigation hyperparameters and model hyperparameters simultaneously to simplify and improve the user experience. The numerical results show that solving the full-space optimization problem is more efficient than focusing on two selected dimensional subspace objectives, and it delivers more insights into the accuracy and fairness trade-offs.

2 Related Work

There has been an explosion of interest in the area of fairness measurement and bias mitigation within the ML community (Mehrabi et al., 2019; Barocas et al., 2019; Garg et al., 2020). Fairness efforts generally focus on three distinct phases: (1) pre-processing (such as data preparation, cleansing, sampling); (2) in-process (model training and tuning); and (3) post-processing (such as calibration and assessment).

Pre-processing methods are applied on the original data directly so as to eliminate biases before fitting any machine learning models. There are studies (Hajian & Domingo-Ferrer, 2012; Hajian, 2013) that discuss methodologies to clean data so that discriminative decision rules will be changed to nondiscriminative ones. Some research suggests not collecting sensitive data to avoid building a discriminative model (Veale & Binns, 2017). Other work such as Calmon et al. (2017) present a framework to probabilistically transform data in order to adjust the unbalances. More pre-processing methods can be found in (Agarwal et al., 2018a; Donini et al., 2018; Kroll, 2015). However, in

general, simply taking actions on the original data alone cannot guarantee that the final model will be nondiscriminative.

In-process methods take fairness into consideration in the training stage of the model-building process. Zafar et al. (2017) address bias mitigation by adding fairness constraints when training models. In their framework, they can choose to maximize accuracy under fairness constraints or maximize fairness under accuracy constraints. The choice of the best model depends upon real situations. Fairness constraints are applied in many settings (Kairouz et al., 2019; Corbett-Davies & Goel, 2018; Agarwal et al., 2018b). On the other hand, fairness can be tackled through single objective optimization where fairness factors are considered while designing the optimization metric (Bechavod & Ligett, 2017; Beutel et al., 2019; Xie et al., 2020; Madras et al., 2019). Unfortunately, there is no universal metric that works for all fairness related problems. The ability to successfully address fairness through single objective optimization, depends on both problem type and metric design.

Post-processing methods try to modify the classification results after the classifiers are trained by adding requirements to account for fairness. A number of studies (Fish et al., 2016; Chouldechova, 2017; Corbett-Davies et al., 2017) propose various methods to deal with the fairness concern after a model is trained. Inevitably, by doing this, the final classification model will be biased mathematically and researchers need a large amount of sensitive data to adjust the model. To make matters worse, there are no standard post-processing methods that can always be applied to address fairness; it is dependent upon the problem and data.

Liu & Vicente (2020) use a stochastic multi-objective optimization in model training to handle two objectives: one for model accuracy and one for fairness. The accuracy/fairness trade-off Pareto front is generated rather than a single solution, and the application of a stochastic approximation-based approach allows the Pareto front to be updated for streaming data cases. However, that approach requires modification of the training algorithm. Cruz et al. (2021) trade off model accuracy and fairness through optimization of model hyperparameters and multiple objectives. Here, a decomposition-based approach to multi-objective optimization is employed, where two objectives are combined into a single scalar output under the assumptions that the Pareto front for the accuracy/fairness trade-off is convex. This approach requires a priori specification of preference among objectives using a weighting parameter.

3 Our Approach

It is understood that AutoML is itself a challenging NP-hard problem. Most previous work is primarily focused on the single objective case with variable bounds. Thus we build on the Autotune framework (Koch et al., 2018; Gardner et al., 2019) which uses a hybrid strategy that combines a genetic algorithm that natively supports multiple objectives paired with direct-search that has been lifted to the higher dimensional objective cases. The approach has been shown to be robust across a number of applications (Koch et al., 2017; Hughes et al., 2020). Gardner et al. (2019) propose a black-box optimization framework that supports multi-level parallel hybrid derivative-free optimization for AutoML problems in the presence of multiple objectives and constraints.

3.1 The Autotune Framework

Autotune is designed to be an AutoML framework that's capable of tuning hyperparameters and architectures for many of the popular machine learning model types: decision trees, forests, gradient boosted trees, neural networks, support vector machines, factorization machines, Bayesian network classifiers, and more. The tuning of models is performed by a hybrid set of search methods that is customizable by the user. Autotune also employs multiple levels of parallelism for efficient execution by parallelizing both model training and model tuning. The following Figure 1 shows a high-level depiction of Autotune's design. Machine learning algorithms define the type of model that will be built while the search methods are responsible for proposing various hyperparameter configurations for consideration. Each unique configuration is stored in a cache to improve performance. The model evaluator component is responsible for training and scoring the models associated with the various hyperparameter configurations. And finally, the search and evaluation manager is charged with coordinating all these activities and ultimately saving the best models.

Autotune makes no assumptions about the objective functions it is optimizing which makes it a perfect candidate for derivative-free optimization techniques to be employed (Taddy et al., 2009; Plantenga, 2009; Gray et al., 2010; Griffin & Kolda, 2010). Derivative-free methods can be used whether derivatives are available or not, so long as the the optimization problem being solved doesn't include a large number of decision variables (Gray & Fowler, 2011). When using optimization techniques to tune the hyperparameters of a machine learning model, there is no objective function structure that can be exploited. In such cases, the objective function is seen as a mapping operator connecting a unique hyperparameter configuration to an objective value. Derivative-free optimization is the perfect choice for such use cases.

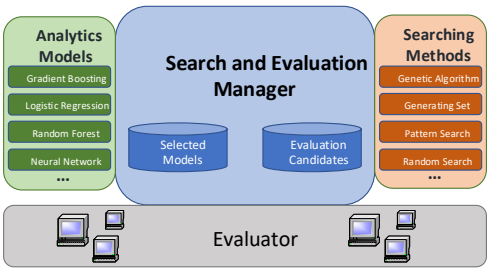

Figure 1: The Autotune framework.

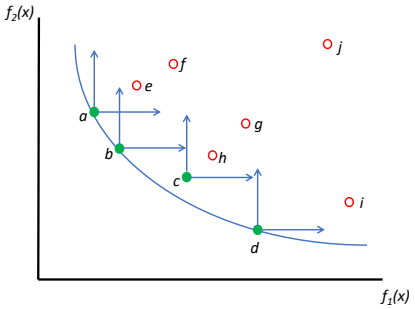

Figure 2: Example Pareto Front.

Autotune's execution is iterative and each iteration repeats the following steps:

1. Search methods submit new points for evaluation
2. Black-box objective functions are called to evaluate each of the new points
3. The computed metrics for each evaluated point are returned back to the search methods

3.2 Multi-Objective Optimization

Multi-objective optimization is a natural fit for tuning the hyperparameters of machine learning models. It's easy to imagine cases where a data scientist would want to build models that maximize accuracy while also minimizing bias. In these types of cases, the desired result is a set of solutions that demonstrate the trade-off between the multiple objectives. The idea is to identify a set of solutions that exhibit the compromise that is necessary between the objectives. In such a set, a given solution could not be improved in any one objective without getting worse in another objective. Identifying this set of solutions is therefore the goal of the multi-objective optimization performed by Autotune.

Mathematically, multi-objective optimization can be defined in terms of *dominance* and *Pareto optimality*. For a k objective minimizing optimization problem, a point (solution) x is *dominated* by a point y if $f_i(x) \geq f_i(y)$ for all $i = 1, \ldots, k$ and $f_j(x) > f_j(y)$ for some $j = 1, \ldots, k$.

A set of nondominated solutions defines a Pareto front. Figure 2 shows a Pareto front with respect to two objectives, $f_1(x)$ and $f_2(x)$ (Gardner et al., 2019). In this example, both objectives are being minimized so the solutions are improving as they move down and left in the figure. The figure shows 10 points that are plotted with respect to the two objectives. In this example, point a dominates $\{e, f, j\}$, b dominates $\{e, f, g, j\}$, c dominates $\{g, h, j\}$, and d dominates $\{i, j\}$. The blue line in the figure depicts the true Pareto front for this example problem. The figure shows how point c has not converged to the true Pareto front yet, but is still not dominated by any of the other points shown. Thus, point c contains points around it that have smaller values of f_1 and f_2 but unfortunately, these points have not been identified yet.

When mitigating model bias and tuning for fairness, Autotune's ability to optimize over multiple objectives becomes extremely important. The default hybrid search strategy in Autotune provides an effective way of conducting multi-objective optimization.

3.3 HYBRID SEARCH STRATEGY

Many common multi-objective optimization strategies focus entirely on the use of metaheuristics (Elsken et al., 2018; Michel et al., 2019; Shenfield & Rostami, 2017). Conversely, the default approach used in Autotune is a hybrid strategy that combines global search (Goldberg, 1989) with an efficient direct local search approach (Griffin et al., 2008). This hybrid strategy allows Autotune to benefit from the strengths of both types of approaches. It begins with a Latin Hypercube Sampling (LHS) to seed the population of points. This initial LHS feeds into a Genetic Algorithm (GA) which searches the solution space for favorable hyperparameter configurations. By using a GA, Autotune is able to solve the multi-objective problem directly and ultimately evolve a Pareto-optimal set of solutions in a single run of the optimization. Other approaches require multiple problems being solved independently. In addition to the GA providing a global search, Autotune performs local searches using a Generating Set Search (GSS) algorithm in promising regions of the search space. This GSS strategy enables Autotune to refine promising solutions and ultimately push the Pareto front of points along. The GSS algorithm is configured to improve objective function values while also reducing the crowding distance of points on the Pareto front. Autotune also uses the averaged Hausdorff distance (Schütze et al., 2012) as a way of measuring convergence. The Hausdorff distance metric is used to compare solutions from one iteration to the next and provide a single metric that indicates how far a Pareto front has progressed in that iteration. If the improvement from iteration to iteration stalls, Autotune will eventually terminate. Algorithm 1 provides a high-level view of the hybrid search strategy used by Autotune.

Algorithm 1 Hybrid Search Strategy in Autotune (with asynchronous parallel evaluations)

Require: Population size n_p, and evaluation budget n_b.

Require: Number of centers $n_c < n_p$ and initial step-size $\hat{\Delta}$ and sufficient decrease $\alpha \in (0, 1)$.

1: Generate and evaluate initial parent-points $\mathcal{P}$ using LHS with $|\mathcal{P}| = n_p$.
2: Populate reference cache-tree, $\mathcal{R}$, with unique points from $\mathcal{P}$.
3: Associate $p \in \mathcal{P}$ with step Δ_p initialized to $\hat{\Delta}$. Let $\mathcal{F}$ denote approximate Pareto front.
4: **while** $(|\mathcal{R}| \leq n_b)$ **do**
5: Select $\mathcal{A} \subset \mathcal{F}$ for local search, such that $|\mathcal{A}| = n_c$.
6: **for** $p \in \mathcal{A}$ **do** ▷ Search along compass directions
7: Set $\mathcal{T}_p = \{\}$. For $e_i \in I$ set $\mathcal{T}_p = \mathcal{T}_p \cup \{p + \Delta_p e_i\} \cup \{p - \Delta_p e_i\}$
8: **end for**
9: Generate child-points $\mathcal{C}$ via crossover and mutations on $\mathcal{P}$. Set $\mathcal{T} = \mathcal{C} \cup_{p \in \mathcal{A}} \mathcal{T}_p$.
10: Evaluate $\mathcal{T} \cap \mathcal{R}$ using fast tree-search look-up on $\mathcal{R}$. Evaluate $\mathcal{T} - \mathcal{R}$ and add to $\mathcal{R}$.
11: Update $\mathcal{P}$ with new generation $\mathcal{C}$ and initial step $\hat{\Delta}$.
12: **for** $p \in \mathcal{A}$ **do**
13: If $|\mathcal{T}_p \cap \mathcal{F}| > 0$, select new $p \in \mathcal{F}$ ▷ Pattern search success
14: Else set $\Delta_p = \Delta_p / 2$ ▷ Pattern search failure
15: **end for**
16: **end while**

Using both global and local search dramatically improves Autotune's robustness. The global search is important because it provides good starting points for the local searches. If the hyperparameter optimization problem happens to be convex, the local search would be sufficient, and the additional overhead added by the global search would be unnecessary. However, if the underlying solution landscape has many local minima, the local search alone could get stuck and fail to find better solutions. Instead of trying to guess which approach is best, Autotune simultaneously runs both global and local searches. The two search strategies run concurrently while sharing computational resources and function evaluations. Autotune can handle integer and categorical hyperparameters using strategies similar to those in Griffin et al. (2011). The approach used by Autotune can be likened to a GA with an additional "growth" step. This "growth" step is where the local search algorithms are used to refine promising points in the GA's population of points. By allocating a small fraction of the total evaluation budget, the local search can refine the fitness score (or objective function value) in the neighborhood around the promising points found by the GA.

3.4 MULTI-LEVEL PARALLELIZATION

When tuning both classical machine learning hyperparameters along with bias mitigation hyperparameters, many hyperparameter settings need to be explored. This process can become computationally expensive which highlights the importance of Autotune's ability to train and score models in parallel in a distributed computing environment. Autotune can simultaneously apply multiple instances of global and local searches in parallel. Given an appropriate number of threads and processors, this strategy can result in run time and solution quality being similar to a scenario where you were able to select the best global and local search combination before execution. Due to the sharing that takes place between the global and local searches, this parallel hybrid approach is much more robust when compared to other hybrid approaches that simply use the result from one search as the starting place for the second method.

Autotune's parallel design is extremely powerful and capable of efficiently using compute grids of any size. When Autotune proposes a new set of hyperparameter values for evaluation, this evaluation is handed off to a worker session that will execute the appropriate scripts to train and score a new ML model. In addition to training and scoring, the new model will be evaluated by a bias-mitigation tool to assess the appropriate fairness metrics. Since each of these script executions is independent, they can be executed in parallel. Autotune manages these parallel evaluations and can effectively scale up to utilize all resources available in a computing environment.

4 EXPERIMENTS AND RESULTS

To evaluate the ability of Autotune to effectively tune machine learning models that are both accurate and fair, we conducted several sets of experiments. Each of these experiments used the same basic setup. A Python script was used to define the objective functions of the hyperparameter optimization performed by Autotune. The Python script uses scikit-learn from Pedregosa et al. (2011) to train a logistic regression model and then applies the Fairlearn package from Bird et al. (2020) to evaluate the new model for bias.

4.1 SCALABILITY OF AUTOTUNE

For our first experiment, we tested how well Autotune scales as the available computing resources increase. As described above, we used scikit-learn to build logistic regression models and the Fairlearn package for assessing fairness metrics when building our models. For this experiment we used the popular Adult data set from the UCI Machine Learning Repository (Dua & Graff, 2017) and Autotune was configured to execute a maximum of 500 objective function evaluations. Figure 3 shows overall runtimes for Autotune on various size computing grids. The figure shows that when Autotune is run in a computing environment with a single worker node, the overall runtime is 4,423 seconds. The runtime is reduced to 127 seconds when using 64 computing nodes. This represents a speed up of about 35x, which is significant considering the computational overhead associated with startup and extra coordination between worker nodes.

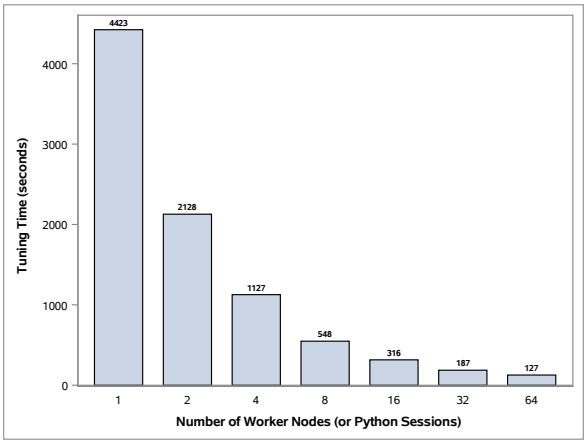

Figure 3: Autotune runtimes with different numbers of worker nodes (or parallel Python sessions). Autotune scales well to efficiently use all available computing resources. In an environment with only 1 worker node, Autotune evaluates objective functions sequentially within a single Python session. When run with 64 worker nodes, Autotune is able to run 64 Python sessions in parallel, allowing for total runtime to be reduced from 4423 seconds to 127 seconds.

4.2 IMPORTANCE OF MULTI-OBJECTIVE TUNING

In this experiment, we demonstrate Autotune's ability to tune with respect to multiple objectives at one time and why this is important. In the "Single Objective Optimization" columns of Table 1 we show the results of running Autotune three different times. Each of these three executions of Autotune is tuning a logistic regression model with respect to a single objective. For this experiment, we are using the German Credit Data from the UCI Machine Learning Repository (Dua & Graff, 2017). Though overall runtime is not considered for this experiment, it's worth noting that for this case, Autotune was configured to execute a maximum of 500 objective function evaluations.

The table shows each tuning objective along the top, and each of the three columns contains the values of the various metrics that were obtained from each execution of Autotune. For instance, if you look at the column labeled "Accuracy", you can see the values that were obtained when we set up Autotune to tune simply for maximizing the accuracy of the model. When run in this way, Autotune was able to find a model with an accuracy of 0.773 (or 77.3%). Further investigation of this model demonstrated a "Demographic Parity" value of 0.027, and an "Error Parity" value of 0.003.

The next column shows the results of executing Autotune a second time. This time Autotune was configured with a single objective to minimize the fairness metric "Demographic Parity". As the table shows, when focused on minimizing demographic parity, Autotune was able to achieve a value of 0 for that fairness metric. However, the model found by Autotune only had an accuracy of 0.7 (or 70%). These results in the "Single Objective Optimization" columns of Table 1 show that when tuning with respect to a single objective, the other metrics of interest can suffer.

The final column in Table 1 shows the results of a single execution of Autotune when configured to optimize all three objectives simultaneously. In this case, the table shows that we are able to achieve the best values for each of the three metrics in a single execution of Autotune. In most real-world scenarios, it is common to be interested in optimizing multiple objectives. When the values of those objectives are at odds with each other, having a system capable of optimizing over multiple objectives at once is the best way to find the best possible values for each of the objectives.

	Single Objective			**Multi-Objective**
	Maximize Accuracy	**Minimize Demographic Parity**	**Minimize Error Parity**	**All 3 Objectives**
Accuracy	0.773	0.700	0.760	0.773
Demographic Parity	0.027	0	0.013	0
Error Parity	0.003	0.028	0	0

Table 1: Power of Tuning Multiple Objectives Simultaneously: The "Single Objective Optimization" columns show the results of each of the three metrics when tuning with respect to each of the single metrics. The diagonal values (0.773, 0, 0) represent the best attainable values for each metric. The"Multi-Objective Optimization" column shows the results when tuning with respect to all three metrics at once. In this case, we obtain the best for all three objectives in one execution of Autotune.

4.3 IMPACT OF FAIRNESS HYPERPARAMETERS

Fairness mitigation algorithms all seek to find a balance point between what are often conflicting objectives – multiple measures of model accuracy and measures of prediction fairness. The algorithms necessarily must surface to the user a set of one or more hyperparameters that can control the strength of the mitigation algorithm. For example, if the reweighing scheme is too strong, the overall accuracy may degrade beyond a point for which the model is no longer useful. On the other hand, if the setting is too weak, the inherent bias may only be negligibly improved, making the model similarly unusable due to unacceptable levels of bias.

Because there is no well-defined optimal single point, mitigation algorithms must surface "steering" parameters that drive the algorithm toward the user's preferred balance point that compromises optimal accuracy versus optimal fairness. Similarly, when we transform the machine learning problem and/or data, the optimal model training hyperparameters also will necessarily change. Thus, the data

scientist, when attempting to mitigate bias, must balance both classical ML model hyperparameters as well as new mitigation hyperparameters simultaneously. This implies a new AutoML class of problems that are inherently multi-objective.

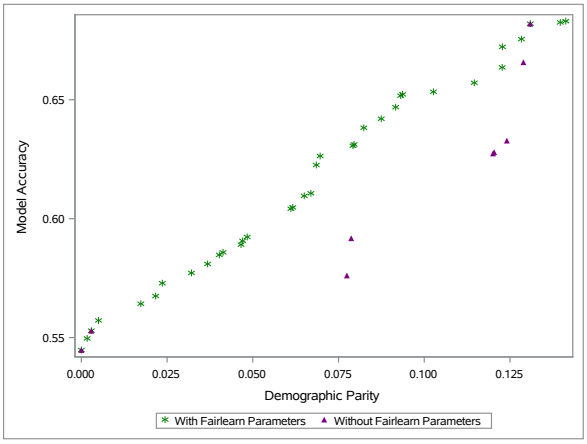

Figure 4: Comparing tuning with and without Fairlearn parameters. Tuning with the COMPAS data set, with respect to two objectives: Accuracy and Demographic Parity. Pink triangles show the Pareto front when tuning only the regression hyperparameters. Green stars show the Pareto front when adding the Fairlearn parameters to the list of hyperparameters being tuned. By adding the Fairlearn parameters we achieve a dominant Pareto front shown by the green stars.

For this experiment, we used the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) Recidivism Risk Score Data (Larson et al., 2016). This data set is commonly used to demonstrate the importance of bias mitigation strategies in machine learning. Figure 4 shows the results of tuning with and without the Fairlearn mitigation hyperparameters as part of the tuning process. Without including the fairness mitigation hyperparameters, the resulting Pareto front is sparse, and is strongly dominated by the Pareto front resulting from an approach that handles both sets of hyperparameters simultaneously.

4.4 Higher Order Multi-objective Tuning

As previously mentioned, the ability to optimize over multiple objectives simultaneously when tuning machine learning models is extremely important. In this experiment, we further demonstrate the power of Autotune and its ability to tune hyperparameters with respect to more than two objectives. For this experiment we used the popular Adult data set from the UCI Machine Learning Repository (Dua & Graff, 2017). The Adult data is curated from census data.

We chose a total of five metrics we would use to measure model quality: accuracy, demographic parity, equalized odds, true positive parity, and false positive parity. We first configured Autotune to build machine learning models, only considering two of the performance metrics at a time. We allowed Autotune to evaluate a total of 500 candidate models before returning the nondominated models that were found. We ran in this configuration four different times, each time optimizing both model accuracy and one of the four Fairlearn metrics. We then ran Autotune a fifth time with the same limit of 500 evaluations, only this time we configured Autotune to optimize over all five of the metrics at the same time.

The plots in Figure 5 show the results from these 5 executions of Autotune. Each of the four plots in Figure 5 shows the trade-off between model accuracy and one of the Fairlearn metrics. The red circles in each of the plots show the results that were obtained when Autotune was solving each of the bi-objective optimization problems. The blue stars in the plots show the Pareto front found by solving the problem with five objectives. In each of these plots, the five-objective Pareto front is projected to the two dimensions shown in the plot.

Since we are attempting to maximize accuracy while minimizing the Fairlearn metrics, the upper left corner of each plot represents the area of the solution space where the trade-off between the two objectives is balanced. All four of the Pareto fronts shown in Figure 5 are similar in that a portion of the Pareto front along the top of the plot is relatively close to flat. These nearly-flat regions indicate solutions where a significant improvement in bias reduction can be achieved for very little loss in model accuracy. Given this shape of each of these Pareto fronts, the upper left portion of each plot represents a promising region for potentially desirable solutions. As an example, looking at the plot showing True Positive Parity vs. Model Accuracy, you can see that more than 20% of true positive

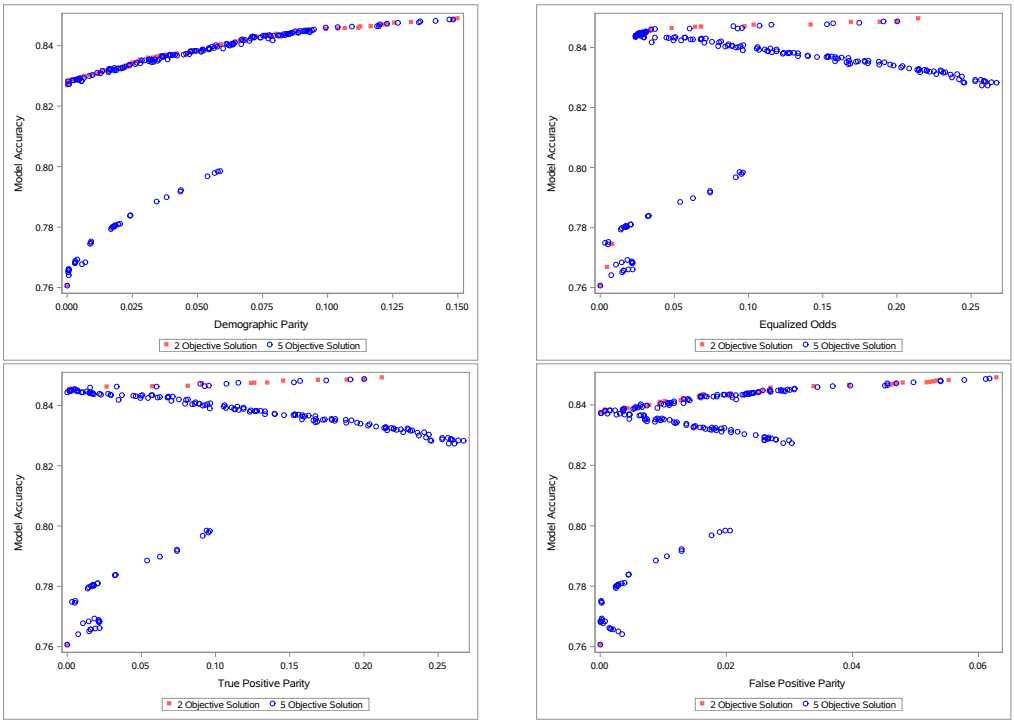

Figure 5: Adult Data: A single Autotune execution with 500 model evaluations is capable of identifying just as good of a solution as four individual Autotune executions with a total of 2000 model evaluations. This clearly shows the strength of being able to tune machine learning models with respect to larger numbers of objectives.

parity difference can be gained by giving up less than 1% of model accuracy. This is the type of insight that can only be gained by tuning such models in a multi-objective setting.

The plots all show that by solving a five-objective optimization problem, we are able to find just as good of a Pareto front in each of these cases. This is significant because by solving the five-objective problem once, Autotune is able to achieve this result in a maximum of 500 model evaluations. However, to achieve the same results for each of the Fairlearn metrics, we had to solve four individual Autotune executions, each using up to 500 evaluations. This means that by optimizing for all five objectives simultaneously, we were able to do just as well as solving four different Autotune problems, except instead of executing 2000 objective function evaluations, we only needed 500 objective function evaluations.

5 Conclusion

In this paper we have described Autotune, an AutoML framework capable of building machine learning models in a way that can efficiently account for both model accuracy and model fairness. With the ever-increasing popularity of bias mitigation strategies in machine learning, a framework like Autotune that is able to build models with a consideration for fairness metrics is extremely timely and important. Autotune is flexible in that it allows for hybrid search strategies to be defined by the user. It is also able to effectively scale based on the computing resources available to it. Through experimentation, we have demonstrated the importance of Autotune's ability to consider multiple objectives simultaneously when tuning a machine learning model. By allowing a user to configure higher numbers of tuning objectives, Autotune is able to build models that exhibit the trade-offs between all the objectives. We have also shown why it is important to include fairness parameters along with machine learning hyperparameters in the tuning process. By tuning both sets of hyperparameters at once, we were able to achieve superior results to those systems that only tune one set of hyperparameters at a time.

REPRODUCIBILITY STATEMENT

The experimental results presented in this paper were obtained using widely available and commonly used data sets (German Credit data set, UCI Adult data set, COMPAS data set), an open source model training routine (sklearn.linear_model.LogisticRegression), an open source fairness measurement and mitigation tool (Fairlearn package described in (Bird et al., 2020)), and a proprietary optimization framework, Autotune. The sensitive attribute studied was gender (male/female) for the German credit data and Adult data set and African_American for the COMPAS data set. The logistic regression model training hyperparameters as well as the Fairlearn bias mitigation hyperparameters and their ranges are given in the Appendix. In part, the contribution of this work includes the development and application of a proprietary optimization framework, Autotune, that supports multi-objective optimization using robust and efficient hybrid search strategies – critical functionality missing from many hyperparameter tuning packages. While the implementation of this framework is not freely available, the algorithm for the default hybrid search strategy is given in this paper. The core search methods – combination of Latin Hypercube Sampling, a multi-objective Genetic Algorithm, and a Generating Set Search algorithm – are available in various literature and coded form (Goldberg, 1989; Griffin et al., 2008; Elsken et al., 2018; Michel et al., 2019; Shenfield & Rostami, 2017). A sample script used by Autotune for the experiments is included in Appendix. More experiment scripts and programs will be publicly available at a later time.

REFERENCES

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudik, John Langford, and Hanna Wallach. A reductions approach to fair classification. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 60–69. PMLR, 10–15 Jul 2018a. URL https://proceedings.mlr.press/v80/agarwal18a.html.

Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna M. Wallach. A reductions approach to fair classification. *CoRR*, abs/1803.02453, 2018b. URL http://arxiv.org/abs/1803.02453.

Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

Yahav Bechavod and Katrina Ligett. Penalizing unfairness in binary classification. *arXiv preprint arXiv:1707.00044*, 2017.

Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John T. Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *CoRR*, abs/1810.01943, 2018. URL http://arxiv.org/abs/1810.01943.

Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Li Wei, Yi Wu, Lukasz Heldt, Zhe Zhao, Lichan Hong, Ed H Chi, et al. Fairness in recommendation ranking through pairwise comparisons. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2212–2220, 2019.

Sarah Bird, Miro Dudik, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in ai. Technical Report MSR-TR-2020-32, Microsoft, May 2020. URL https://www.microsoft.com/en-us/research/publication/fairlearn-a-toolkit-for-assessing-and-improving-fairness-in-ai/.

Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/9a49a25d845a483fae4be7e341368e36-Paper.pdf.

Joymallya Chakraborty, Tianpei Xia, Fahmid M. Fahid, and Tim Menzies. Software engineering for fairness: A case study with hyperparameter optimization. *CoRR*, abs/1905.05786, 2019. URL http://arxiv.org/abs/1905.05786.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*, 2018.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*, pp. 797–806, 2017.

André F. Cruz, Pedro Saleiro, Catarina Belém, Carlos Soares, and Pedro Bizarro. Promoting fairness through hyperparameter optimization. abs/2103.12715, 2021. URL https://arxiv.org/pdf/2103.12715.pdf.

Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In Marc Schoenauer, Kalyanmoy Deb, Günther Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel (eds.), *Parallel Problem Solving from Nature PPSN VI*, pp. 849–858, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg. ISBN 978-3-540-45356-7.

Michele Donini, Luca Oneto, Shai Ben-David, John Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. *arXiv preprint arXiv:1802.08626*, 2018.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Efficient multi-objective neural architecture search via lamarckian evolution, 2018. URL https://arxiv.org/pdf/1804.09081.pdf.

Virginia Eubanks. *Automating Inequality: How High-Tech Tools Profile, Police, and Punish the Poor*. St. Martin's Press, Inc., USA, 2018. ISBN 1250074312.

Benjamin Fish, Jeremy Kun, and Ádám D Lelkes. A confidence-based approach for balancing fairness and accuracy. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pp. 144–152. SIAM, 2016.

Steven Gardner, Oleg Golovidov, Joshua Griffin, Patrick Koch, Wayne Thompson, Brett Wujek, and Yan Xu. Constrained multi-objective optimization for automated machine learning. In Lisa Singh, Richard D. De Veaux, George Karypis, Francesco Bonchi, and Jennifer Hill (eds.), *2019 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2019, Washington, DC, USA, October 5-8, 2019*, pp. 364–373. IEEE, 2019. doi: 10.1109/DSAA.2019.00051. URL https://doi.org/10.1109/DSAA.2019.00051.

Pratyush Garg, John D. Villasenor, and Virginia Foggo. Fairness metrics: A comparative analysis. In Xintao Wu, Chris Jermaine, Li Xiong, Xiaohua Hu, Olivera Kotevska, Siyuan Lu, Weija Xu, Srinivas Aluru, Chengxiang Zhai, Eyhab Al-Masri, Zhiyuan Chen, and Jeff Saltz (eds.), *IEEE International Conference on Big Data, Big Data 2020, Atlanta, GA, USA, December 10-13, 2020*, pp. 3662–3666. IEEE, 2020. doi: 10.1109/BigData50022.2020.9378025. URL https://doi.org/10.1109/BigData50022.2020.9378025.

David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., 1st edition, 1989. ISBN 0201157675.

G. A. Gray and K. R. Fowler. The effectiveness of derivative-free hybrid methods for black-box optimization. *International Journal of Mathematical Modeling and Numerical Optimization*, 2: 112–133, 2011.

G. A. Gray, K. R. Fowler, and J. D. Griffin. Hybrid optimization schemes for simulation-based problems. *Procedia Computer Science*, 1:1349–1357, 2010.

J. D. Griffin and T. G. Kolda. Asynchronous parallel hybrid optimization combining direct and gss. *Optimization Methods and Software*, 25:797–817, 2010.

J. D. Griffin, T. G. Kolda, and R. M. Lewis. Asynchronous parallel generating set search for linearly constrained optimization. *SIAM Journal on Scientific Computing*, 30:1892–1924., 2008.

J. D. Griffin, K. R. Fowler, G. A. Gray, and T. Hemker. Derivative-free optimization via evolutionary algorithms guiding local search (eagls) for minlp. *Pacific Journal of Optimization*, 7:425–443, 2011.

Sara Hajian. Simultaneous discrimination prevention and privacy protection in data publishing and mining. *arXiv preprint arXiv:1306.6805*, 2013.

Sara Hajian and Josep Domingo-Ferrer. A methodology for direct and indirect discrimination prevention in data mining. *IEEE transactions on knowledge and data engineering*, 25(7):1445–1459, 2012.

M. Hicks. *Programmed Inequality: How Britain Discarded Women Technologists and Lost Its Edge in Computing*. History of Computing. MIT Press, 2017. ISBN 9780262342940. URL https://books.google.com/books?id=pzATDgAAQBAJ.

Ed Hughes, Steve Gardner, Josh Griffin, and Oleg Golovidov. The new solveblackbox action in sas optimization 8.5. In *SAS Global Forum 2020 Conference*, Cary, NC, 2020. SAS Institute Inc.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Patrick Koch, Brett Wujek, Oleg Golovidov, and Steven Gardner. Automated hyperparameter tuning for effective machine learning. In *SAS Global Forum 2017 Conference*, 2017.

Patrick Koch, Oleg Golovidov, Steven Gardner, Brett Wujek, Joshua Griffin, and Yan Xu. Autotune: A derivative-free optimization framework for hyperparameter tuning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pp. 443–452, 2018.

Joshua Alexander Kroll. *Accountable algorithms*. PhD thesis, Princeton University, 2015.

J. Larson, S. Mattu, and J. Angwin. Compas data on github, 2016. URL https://github.com/propublica/compas-analysis.

Suyun Liu and Luís Nunes Vicente. Accuracy and fairness trade-offs in machine learning: A stochastic multi-objective approach. *CoRR*, abs/2008.01132, 2020. URL https://arxiv.org/abs/2008.01132.

David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Fairness through causal awareness: Learning causal latent-variable models for biased data. In *Proceedings of the conference on fairness, accountability, and transparency*, pp. 349–358, 2019.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *CoRR*, abs/1908.09635, 2019. URL http://arxiv.org/abs/1908.09635.

Guillaume Michel, Mohammed Amine Alaoui, Alice Lebois, Amal Feriani, and Mehdi Felhi. DVOLVER: Efficient pareto-optimal neural network architecture search, 2019.

Safiya Umoja Noble. *Algorithms of Oppression: How Search Engines Reinforce Racism*. NYU Press, 2018. ISBN 9781479849949. URL http://www.jstor.org/stable/j.ctt1pwt9w5.

Cathy O'Neil. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group, USA, 2016. ISBN 0553418815.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

T. Plantenga. Hopspack 2.0 user manual (v 2.0.2). Technical report, Sandia National Laboratories, 2009.

Pedro Saleiro, Benedict Kuester, Abby Stevens, Ari Anisfeld, Loren Hinkson, Jesse London, and Rayid Ghani. Aequitas: A bias and fairness audit toolkit. *CoRR*, abs/1811.05577, 2018. URL `http://arxiv.org/abs/1811.05577`.

O. Schütze, X. Esquivel, A. Lara, and C. A. Coello Coello. Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 16:504–522, 2012.

Alex Shenfield and Shahin Rostami. Multi-objective evolution of artificial neural networks in multi-class medical diagnosis problems with class imbalance. In *CIBCB*, pp. 1–8. IEEE, 2017.

M. A. Taddy, H. K. H. Lee, G. A. Gray, and J. D. Griffin. Bayesian guided pattern search for robust local optimization. *Technometrics*, 51:389–401, 2009.

Michael Veale and Reuben Binns. Fairer machine learning in the real world: Mitigating discrimination without collecting sensitive data. *Big Data & Society*, 4(2):2053951717743530, 2017.

Ning Xie, Gabrielle Ras, Marcel van Gerven, and Derek Doran. Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*, 2020.

Bo Xiong, Yimin Huang, Hanrong Ye, Steffen Staab, and Zhenguo Li. Mofa: Modular factorial design for hyperparameter optimization. 2021.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pp. 962–970. PMLR, 2017.

Eckart Zitzler, Marco Laumanns, Lothar Thiele, Carlos M. Fonseca, and Viviane Grunert da Fonseca. Why quality assessment of multiobjective optimizers is difficult. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, GECCO'02, pp. 666–674, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1-55860-878-8.

A Appendix

A.1 Objective function description

In our approach we used the Fairlearn package described Bird et al. (2020) implementing exponential gradient reduction and grid search described in Agarwal et al. (2018b). The latter approach was recommended for the case when the number of objectives (or constraints) is small. In the latter case, Bird et al. (2020) treat the problem as multi-objective, provided the user code to generate a Pareto front after their algorithm exits. The grid search algorithm itself generates a set of Lagrange multipliers $\Lambda = \{\lambda_1, \ldots, \lambda_P\}$ with $\lambda_i \in \mathbb{R}^m$, where m denotes the dimensions of the fairness constraints and P denotes the number of models the user is willing to train. They then transform the underlying training problem for each $\lambda_i \in \Lambda$ (as recommended by Agarwal et al. (2018b)) and proceed to generate P models that vary in both accuracy and fairness.

As mentioned earlier, the purpose of this paper is not to access the strength or promote a particular approach. Rather, we seek to provide a mechanism to incorporate any fairness mitigation hyperparameters surfaced by developers to improve models in a multi-objective sense, including those mitigating bias. In terms of Beutel et al. (2019) they have effectively created a map $F(\lambda)$ with $F : \mathbb{R}^m \rightarrow \mathbb{R}^k$ where m denotes the dimension of λ (e.g. the number of constraints and k denotes the number of objectives to include (but not necessarily limited to) accuracy and corresponding fairness metrics as defined by Bird et al. (2020).

Note covered in Bird et al. (2020) is the observation that it is recommended to tune model training hyperparameters whenever the training data changes. Of course this might be done beforehand, however, this implies that the fairness mitigation transformations are all similar, which is not guaranteed for more complex pipelines. Further, there is no reason that tuning the models itself does not have some impact on the fairness scores as well as accuracy Chakraborty et al. (2019). Thus it makes sense to capture intermediate models that may be produce while tuning model hyperparameters and fairness hyperparameters simultaneously. Thus we define the composite AutoML and fairness multi-objective optimization problem as

$$\min_{\lambda \in \mathbb{R}^m \text{ and } h \in \mathbb{R}^n} \quad F(\lambda, h) \tag{1}$$

$$\text{subject to} \quad \lambda_\ell \leq \lambda \leq \lambda_u, \tag{2}$$

$$h_\ell \leq h \leq h_u,$$

where $F : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^k$, λ is used as in Bird et al. (2020) during grid search, and h corresponds to typical optimization and model hyperparameters tuned during training to improve accuracy on a given test or validation set. Here $[\lambda_\ell, \lambda_u]$ and $[h_\ell, h_u]$ denotes corresponding bounding boxes containing promising regions of improvement.

Note that rather than tuning the hyperparameters of the grid-search algorithm described in Agarwal et al. (2018b) itself, we directly tune the same hyperparameters as their proposed algorithm. There are a couple of reasons this is preferable. First our approach starts with a Latin Hypercube Sampling (LHS), that should be superior to a simple grid-search and applicable to even greater dimensions. Second, for in-processing type mitigation algorithms they often must train many models to obtained a preferred set of solutions. It thus makes sense to work at a lower level so that we can measure and assess every model we trained for quality and ensure all Pareto points are kept and leveraged. Thus our $F(\lambda, h)$ is in this case tuning the same hyperparameters as the corresponding grid-search algorithm with respect to the λ variables. This of course will be far more efficient than tuning what is arguably also a tuning algorithm. A final observation is that our algorithm makes no distinctions currently between the mitigation hyperparameters λ and the classical hyperparameters denoted by h; that is we apply Algorithm 1 to the problem

$$\min_{v \in \mathbb{R}^{n+m}} \quad F(v)$$

$$\text{subject to} \quad v_\ell \leq v \leq v_u,$$

,

where $v = (\lambda, h)$, $F(\lambda, h)$ is redefined simply as $F(v)$, $v_\ell = (\lambda_\ell, h_\ell)$, and $v_u = (\lambda_u, h_u)$. In Figure 6 we demonstrate the power of a sophisticated multiobjective algorithm using an algorithm such as grid-search as essentially a starting point.

A.2 Value of GA and GSS When Tuning

As previously described, Autotune uses a hybrid search strategy that includes a Latin Hypercube Sample (LHS), a Genetic Algorithm (GA), and a Generating Set Search (GSS) all working together to find the best possible solutions. Many hyperparameter tuning systems are doing things like random search or grid search and we have become convinced through experimentation that the GA and GSS are vital components of Autotune, especially when considering a multi-objective setting with various types of hyperparameters being tuned. Figure 6 illustrates the significant improvement that can be realized by running the full hybrid search strategy with the GA and GSS. In the figure, the 2 red circles represent the Pareto front found by simply running Autotune's initial iteration which involves evaluating the solutions proposed by the LHS. Those solutions are completely dominated by the green diamonds which are the solutions identified by allowing the full Autotune hybrid search strategy to execute together.

A.3 Variable descriptions and bounds used

In this section we list and describe the variable bounds used for both mitigation and model tuning parameters. While the approach is not specific to any model type as it is largely a black-box optimization approach, for all examples we used logistic regression as a base model type. For the logistic regression we tuned the following model and mitigation hyperparameters:

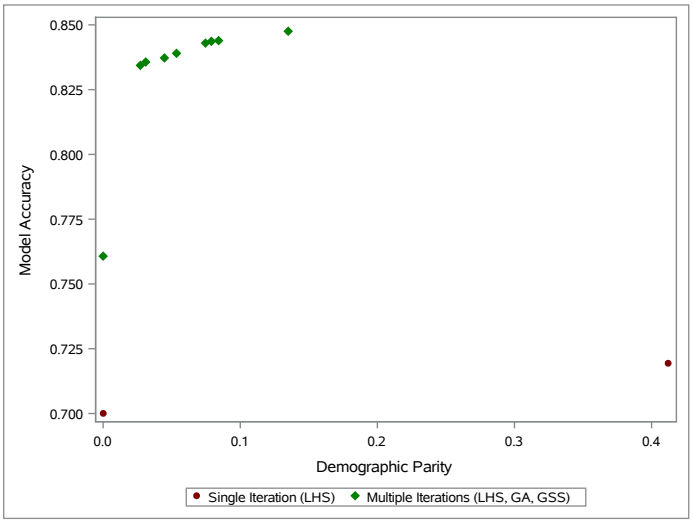

Figure 6: Comparing tuning with and without GA and GSS. Tuning with the COMPAS data set, with respect to 2 objectives: Accuracy and Demographic Parity. Red circles show the solutions found by simply running Autotune's initial LHS. Green diamonds are the solutions found by running the full hybrid search strategy in Autotune (LHS, GA, GSS).

Description	**Type**	**Range**	**Role**
Optimality tolerance	float	[1e-10 .1]	model tuning
Regularization strength	float	[1e-12, 100]	model tuning
Solver type	categorical	{ liblinear, saga, lbfgs, newton-cg }	model tuning
Use Bias/Fit-intercept	categorical	{true, false}	model tuning
Number of CPU cores	integer	[1,6]	model tuning
Lagrange multipliers	float vector	[0,2e]	mitigation

Table 2: Hyperparameter Variable Description. In this table we specify the hyperparameters used to perform multi-objective optimization and their corresponding range, type, and role. The first five are primarily used when perform classic model tuning to improve accuracy. We include the number of cores as this can affect the solution found and hence the corresponding objective values. The last vector of hyperparameters represents the mitigation hyperparameters used by Agarwal et al. (2018b) to perform grid search over the space of Lagrange multipliers. The dimension is typically small but depends on the definition of the constraints formulated by Bird et al. (2020).

Those hyperparamters seem most sensitive with respect to fairness and accuracy metrics. As mentioned earlier, we use directly the Lagrange multipliers λ as defined in Agarwal et al. (2018b) using the hypercube $[0, 2e]$ as corresponding bounds. Internally λ is used to generate a weight vector as well as a modification to the target variable for the underlying training algorithm. As is typical in AutoML, the fairness and accuracy metrics are computed with respect to a test set to reduce the chance of over-training with respect to the given hyperparameters.

A.4 Sample Script

The following is a sample script that was used to execute the Autotune system and tune a logistic regression model. In this example, we are tuning a total of 9 hyperparameters: 5 model-related and 4 pertaining to fairness. The script defines the hyperparameters being tuned (which are presented as decision variables to the optimization solver) and also indicates the list of 5 objectives that we wish for the solver to consider when solving the problem. This example specifies the name of a shell script (fairObj.sh) that it wants the optimizer to use as the definition of its black-box objective function. This particular script calls Python code that uses scikit-learn to train and score the model and then uses the Fairlearn package to assess the model's fairness with respect to sensitive attributes.

```
optimization.solveBlackbox /
/* Define decision variables for the hyperparameter optimization */
decVars = {
        {name="vartol",      type='C', lb=1e-10, ub=.1}  /* optimality tolerance */
```

```
        {name="varc",        type='C', lb=1e-12, ub=100} /* inverse regularization */
        {name="varisolve",   type='I', lb=0,     ub=3}   /* solver type */
        {name="varibias",    type='I', lb=0,     ub=1}   /* use bias/fit-intercept */
        {name="varnjobs",    type='I', lb=1,     ub=6}   /* cpu cores */
        {name="varlam1",     type='C', lb=0,     ub=2}   /* lagrange multiplier 1 */
        {name="varlam2",     type='C', lb=0,     ub=2}   /* lagrange multiplier 2 */
        {name="varlam3",     type='C', lb=0,     ub=2}   /* lagrange multiplier 3 */
        {name="varlam4",     type='C', lb=0,     ub=2}   /* lagrange multiplier 4 */
}

/* Define the 5 objectives for the optimization problem */
obj = {
        {name='acc',      type='max'},  /* model accuracy */
        {name='dempar',   type='min'},  /* demographic parity */
        {name='eodds',    type='min'},  /* equalized odds */
        {name='tpospar',  type='min'},  /* true positive parity */
        {name='fpospar',  type='min'}   /* false positive parity */
}

/* Define the objective function that will be called by the
 optimizatino solver. Here we pass the name of shell script
 that calls the appropriate Python script to train and score
a model and then assess the model for fairness.
*/
func = {eval="fairObj.sh"}

/* Define optimization parameters */
maxfunc   = 500
popsize   = 100
 nparallel  = 64

/* Specify output tables for saving results */
cacheOut  = {name="cacheOut", replace=true},
primalOut = {name="primalOut", replace=true};
;
```