Experimental analysis of multi-step pipelines for fair classifications – More than the sum of their parts?

1st Nico Lässig
University of Stuttgart, IPVS
Stuttgart, Germany
nico.laessig@ipvs.uni-stuttgart.de

2nd Melanie Herschel
Nanyang Technological University
Singapore
melanie.herschel@ntu.edu.sg

***Abstract*—The problem of biased machine learning predictions has led to many alternative approaches to mitigate the problem. They are typically studied and evaluated by focusing on the input data, the trained model, and the performance of the model predictions. We take a broader perspective, considering approaches in the context of a multi-step pipeline. We study fair classification in a pipeline comprising multiple data preparation steps, parameter optimization, and three types of approaches (pre-, in-, and post-processing) designed to reduce bias that may be applied consecutively. This pipeline leads to a trained model to be evaluated in terms of quality (e.g., accuracy) and fairness.**

We experimentally evaluate the effect differently combined implementations of the pipeline components have on the performance of more than 40 fairness-inducing algorithms. Key findings made possible by this pipeline perspective include: (1) Choosing a bias reducing algorithm greatly simplifies when implementing suited data preparation or parameter optimization, as the difference in performance between methods shrinks, making almost any choice a good one. (2) Several component or pipeline implementations often assumed to have positive or negative effects on performance prove to have little or even contrary effects to the expectations. (3) While many approaches have been published for fair classification in the last decade and shown to improve on previous solutions in specific settings, our broad analysis reveals a stagnating performance trend.

Our analysis shows that synergetic effects between pipeline components need to be carefully taken into account for further research on fair end-to-end data processing. It further raises the more fundamental question of how the study of the problem evolves, both in terms of proposed solutions and benchmarking.

***Index Terms*—fairness, machine learning pipelines, evaluation**

I. Introduction

Today, many companies and systems rely on machine learning (ML) to predict various events and recommend actions that should be taken. In many scenarios, it is critical for predictions to be fair and not to exhibit bias towards specific groups of people. Example scenarios are credit scoring [1], hiring systems [2], college admissions [3], or crime recidivism [4]. Yet, ML models revealed to be discrimative in diverse settings, e.g., exhibiting gender bias in recruitment tools [5] or job advertising [6] or race bias in crime forecasting [7].

The above examples illustrate the large variety of data-driven applications that require the incorporation of fairness to not discriminate people from a certain protected group (e.g., based on a specific gender, race, or a combination of both). A first step, already incorporated in some laws, is to entirely disregard protected attributes, such as gender or race, during data processing, e.g., for training ML models. In addition, research has explored algorithmic solutions for fair clustering [8], fair ranking [9], [10], and fair regression [11], among others. Yet, most research has focused on fair classifications [11]–[14], with a large majority concentrating on binary classification. A binary classification classifies tuples into two possible classes. One class has the favorable outcome (e.g., a person gets a loan), while the other is the unfavorable class (e.g., a person is denied a loan). *This paper focuses on the experimental evaluation of fairness-inducing algorithms for binary classification, as it has high practical relevance and, given the plethora of available solutions, insights into when or how to use these algorithms are most valuable.*

The algorithms to mitigate bias in binary classifications broadly fall into three categories: *pre-processing*, *in-processing*, and *post-processing* [11]–[16], depending on which stage of the training process they apply to. Some of these fairness-inducing approaches for binary classifications further restrict to situations with a single binary protected attribute, thereby requiring some "binarization" before being applicable to a dataset with multiple or non-binary protected attributes. Binarization may lead to discrimination against individuals with multiple protected attributes [17], [18]. Many of the proposed algorithms also rely on parameters that may have a varying impact on the overall performance. Similarly, removing protected attributes as part of data preparation may impact the overall performance [19], [20]. In practice, further "standard" data preparation steps such as sampling or dimensionality reduction precede ML model training, which may also affect the model performance. *This paper evaluates fair classification algorithms in the context of a processing pipeline that comprises several data preparation steps, parameter optimization for fair classification algorithms, and the three possible stages of pre-, in-, and post-processing, coupled with model training.* This goes beyond earlier work on comparative evaluation, focusing on fair classification algorithms with their input and output "in isolation". While this paper focuses on a few variations of a baseline data processing pipeline, the general methodology and publicly available code base [21] can serve as a blueprint for further valuable experimental studies.

Contributions. Our overarching contribution are *novel insights into the relevance and the interplay of pipeline com-*

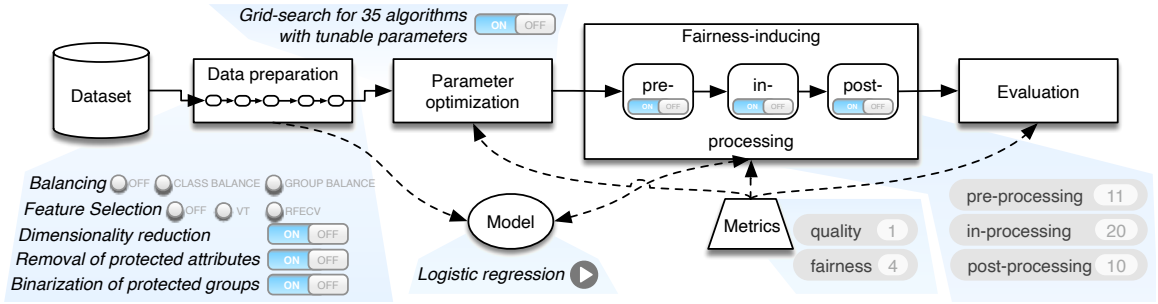


Fig. 1: Processing pipeline overview with implementation options for different components

ponents for fair binary classification tasks. Fig. 1 outlines the general pipeline we consider and summarizes which "implementations" of each component we focus on. Our insights are based on an extensive experimental evaluation of over 40 fairness-inducing algorithms on six real-world datasets, integrated into various pipeline configurations. To the best of our knowledge, this is the most extensive evaluation of such algorithms compared to previous experimental studies and the only one taking a broader and systematic pipeline perspective. Indeed, Hort et al. [22] evaluate the algorithms implemented via AIF360 [23], an established framework for fair classifications. Friedler et al. [24] compare four bias mitigation algorithms. In their work, Chen et al. [25] and Islam et al. [16] consider 10 (resp. 9) algorithms of the AIF360 framework and two (resp. four) additional fair approaches. Guha et al. [26] evaluate the impact of data cleaning on the fairness of datasets, differing from our data preparation focus. Biswas et al. [27] study the fairness impact of specific data preparation pipelines gathered from prior work when combined with standard, non-fairness-inducing classifiers. Selected key insights of our experiments are the following:

(1) Flexible optimization focus. Either using automatically optimized parameters or optimally selected data preparation steps prior to running a bias-mitigating algorithm generally improves the performance (i.e., reduce bias with little impact on accuracy). This, by itself, is not surprising. More interestingly, we observe that data preparation optimization typically is the better choice, and combining both rarely proves necessary. From a practical perspective, this is relevant as it allows competitive performance improvements for algorithms lacking parameters to optimize or may offer a scalable alternative to algorithms with a large parameter search space.

(2) Easy choice of fairness-inducing algorithm. Once we boost performance based on optimized data preparation or algorithm parameters (see Contribution (1)), we observe *no* significant difference between different types of fairness-inducing algorithms, refuting an earlier result [16]. This simplifies the task of choosing among the many bias-mitigating algorithms (no choice is a bad choice), as long as some optimization on prior pipeline components is performed.

(3) No synergy between fairness-inducing algorithms. While it may be tempting to combine pre-, in-, and post-processing techniques to tackle bias from all its angles, we observe that combining multiple approaches often has a negative effect. We attribute this behavior to pre-processing algorithms manipulating the datasets, resulting in inaccurate training data.

(4) Stagnation of bias-vs-accuracy trade-off. Taking into account a large number of bias-mitigating algorithms over a large variety of benchmark settings proposed over more than a decade, and considering both bias and accuracy as equally important, we observe that the performance has been stagnating. This by no means signifies that individual methods may not be better than others in some specific settings. This, however, raises questions about what future research in that area aims for, what (new) methods may be needed, or how such solutions should be benchmarked in the future.

(5) Scalability as important factor. The choice of fairness-inducing algorithm, while simplified through optimization options, depends on requirements on memory or runtime scalability. Our extensive evaluation identifies the algorithms to favor in presence of such requirements and refutes a previous result [16] as we do not observe clear scalability differences between pre-, in-, and post-processing techniques.

Structure. Sec. II summarizes the pipeline configurations our evaluation considers. Sec. III and Sec. IV introduce the fair classification algorithms and metrics we focus on. Sec. V presents our experimental analysis. Sec. VI concludes.

II. Data Processing Pipeline Overview

Fig. 1 summarizes the processing pipeline framework for fair binary classifications that we consider and the component implementations our experimental study covers.

A. Pipeline framework components

The pipeline input is a *dataset*, split into training and validation data. We further need to specify performance *metrics*.

The *data preparation* component allows the integration of data manipulations and transformations common when preparing data for ML training, e.g., projecting out certain attributes, discretizing values, performing dimensionality reduction, etc. These data preparation steps are generally orthogonal to data transformations specifically targeted towards fair classification (which are the focus of dedicated pre-processing algorithms). However, projecting out protected attributes usually does impact fairness, so there is some overlap between data preparation for ML in general and data preparation for fair ML.

The *parameter optimization* component targets the optimal setting of fairness techniques, depending on the (prepared) data

and the considered fairness metric. Parameter optimization is widely used for "accuracy-centered" ML [28]. Standard parameter optimization techniques include grid search, random search [29], or Bayesian optimization [30]. Focusing on fairness, the few algorithms (or evaluations thereof) [22], [24] that apply parameter optimization showcase a gap between good and bad configurations. We integrate parameter optimization as an integral part of a fair classification pipeline.

The *fairness-inducing* component has three sub-components corresponding to different types of algorithms (pre-, in-, and post-processing) for fair classification. In principle, these can be applied in sequence. Most existing algorithms target specific fairness metrics but are amenable to supporting others. Sec. III covers the implemented algorithms in more detail. Some approaches require an additional *model*. For instance, the pre-processing algorithms mutate or return sample weights for the training data. This requires an additional classifier that uses the adapted training data as input.

The *evaluation* component assesses the trained model in terms of classification quality and fairness by applying it to the test data. Used evaluation *metrics* are, in principle, independent from those specified to be the focus of data processing for model training (usually accuracy and a specific fairness metric). The component also serves to assess scalability.

B. Implemented pipeline configurations

Our pipeline implementation focuses on five data preparation steps. *Balancing* based on sampling, *feature selection*, and *dimensionality reduction* represent data preparation steps that are commonly used in feature engineering. In selecting implementation variants for these steps, we reviewed what is common practice in publicly availalbe ML pipelines. For sampling, we use a combination of *SMOTE* oversampling and *Tomek links* undersampling [31]. Based on this sampling strategy, we consider two balancing options. The first, named *class balancing*, balances the label classes. The second, called *group balancing*, is an adaptation that balances all group combinations, including protected attributes and labels. For the feature selection component, we offer two options: The *variance threshold* (short: VT) approach [32] as an unsupervised feature selection technique and the supervised *recursive feature elimination with cross-validation* (short: RFECV) [33] approach. For dimensionality reduction, we use the *principal component analysis* (short: PCA) approach with the automatic choice of dimensionality by Minka [34]. We also consider two data preparation steps that are frequent when focusing on the fairness of trained models. The first such step considers the potential *removal of protected attributes*, as these can either be kept or projected out for the training step. Note they are still kept as metadata, as algorithms typically require them for their fairness-inducing mechanisms. We name the second step *binarization*: if the input data contains multiple protected attributes or a non-binary protected attribute, this step transforms the data to include only a singly binary protected attribute, an expected input to many fair classification algorithms. For instance, when we have two protected attributes {race, sex}, we form one single privileged group (e.g., {white, male}), while all other combinations form the discriminated group.

In our implementation, we can either activate or deactivate parameter optimization. When activated, we opt for *grid search* to find an optimal parameter setting. It builds a grid of all parameters to be tuned and the designated values that we want to observe. Then, the algorithm to be optimized is run for all different combinations of parameter values, returning the parameter values that yield the optimal results. To evaluate the results, a loss function $\hat{L}$ is used that determines the quality of each parameter set. As the goal is to achieve fairness while retaining high accuracy, we choose a loss function that equally weighs the error rate and the considered fairness metric. When deactivating parameter optimization, the algorithms run using their default parameters. We either obtain default settings from the official implementation or, in lack thereof, the best settings of the experiments in the original paper. We choose grid search over more efficient parameter optimization techniques as it offers a systematic way to reach our main goal of assessing algorithm performance through optimized parameter settings compared to the default ones.

The sub-components of the fairness-inducing component can be implemented by choosing among 11 pre-processing, 20 in-processing, and 10 post-processing algorithms (introduced in Sec. III), either separately or in combination. Note, however, that not every algorithm from one type is compatible with every algorithm of another type. For instance, post-processing algorithms may adjust a specific type of classifier not supported by in-processing techniques.

When an algorithm requires an additional model (as explained in Sec. II-A), we train a *Logistic Regression (LR)* classifier [35]. In the experiments by Islam et. al. [16], LR consistently finished among the top. Furthermore, LR is a common choice for experiments of bias mitigation techniques [14] used to verify the models in the original research papers or in the documentation of their implementation.

For each pipeline configuration, our implementation can target one of four fairness metrics (covered in Sec. IV) coupled with a quality metric. Additionally considering all the datasets underlying our evaluation results in over 10.000 pipeline configurations that we ran and analyzed. Our framework and implementation [21] are extensible to further data processing steps, component implementations, and fairness metrics, offering a springboard for new follow-up research.

III. Fair Classification Algorithms

Fair classification approaches are typically categorized based on the processing stage to which they apply [11]–[14]. (1) *Pre-processing* techniques modify the training data or assign different weights to the samples. (2) *In-processing* approaches induce fairness during training. (3) *Post-processing* strategies adjust the input, classifiers, or classifier output.

Tab. I summarizes the approaches we consider in our evaluation and briefly review in Sec. III-A. Sec. III-B describes their features we surveyed and that are pertinent to our evaluation.

	Ref.	Algorithm	Year	Metric	Tuned Parameters	Multiple	Ignore Protected	Opt. Strategy
pre	[36]	Reweighing	2012	dp	-	no	implementation-based	●
	[37]	LFR	2013	dp, consistency	3 (2x continuous, 1x discrete)	no	implementation-based	⋆
	[38]	DisparateImpactRemover	2015	dp	1 (continuous)	no	built-in	●
	[39]	PFR	2019	dp, eod, consistency	1 (continuous)	yes	implementation-based	●
	[40]	Fair-SMOTE	2021	dp, eod, eop	2 (continuous)	yes	implementation-based	●
	[41]	FairSSL-Lx	2022	dp, eod, eop	4 (2x continuous, 1x discrete, 1x binary)	yes	implementation-based	●
	[41]	FairSSL-xT	2022	dp, eod, eop	4 (2x continuous, 1x discrete, 1x binary)	yes	implementation-based	-
	[42]	FS	2022	dp	1 (discrete)	no	built-in	●
	[43]	LTDD	2022	dp, eod, eop	-	no	built-in	●
	[44]	iFlipper	2023	consistency	1 (continuous)	no	built-in	-
	[45]	FairRR	2024	dp, eop, other	1 (continuous)	no	implementation-based	⋆
in	[46]	PrejudiceRemover	2012	dp	1 (continuous)	no	-	●
	[47]	SquaredDifferenceFairLogistic	2017	treq	1 (continuous)	no	built-in	-
	[48]	FairnessConstraintModel	2017	dp	1 (continuous)	no	built-in	-
	[49]	DisparateMistreatmentModel	2017	treq	1 (continuous)	no	built-in	●
	[50]	ConvexFrameworkModel	2017	dp, other indiv.	2 (1x continuous, 1x discrete)	no	built-in	⋆
	[51]	HSICLinearRegression	2017	other	1 (continuous)	no	built-in	-
	[52]	GerryFairClassifier	2018	other	1 (continuous)	yes	-	●
	[53]	AdversarialDebiasing	2018	dp, eod, eop	1 (continuous)	no	built-in	-
	[54]	ExponentiatedGradientReduction	2018	dp, eod	3 (2x continuous, 1x binary)	yes	parameter-based	●
	[54]	GridSearchReduction	2018	dp, eod	2 (1x continuous, 1x binary)	yes	parameter-based	-
	[55]	MetaFairClassifier	2019	dp, other	1 (continuous)	no	-	●
	[56], [57]	AdaFair	2019	eod, treq, dp*, eop*	2 (1x continuous, 1x discrete)	no	-	●
	[58]	FAGTB	2019	dp, eod	3 (2x continuous, 1x discrete)	no	implementation-based	⋆
	[59]	HGR	2019	eod	2 (continuous)	no	built-in	-
	[60]	FairDummies	2020	eod	3 (continuous)	no	built-in	⋆
	[61]	GeneralFairERM	2020	eod	1 (continuous)	no	built-in	⋆
	[62]	MultiAdversarialDebiasing	2021	dp, eod	1 (continuous)	no	built-in	-
	[63]	GradualCompatability	2022	other	3 (2x continuous, 1x discrete)	no	built-in	-
	[64]	FairGeneralizedLinearModel	2022	eod, other	2 (1x continuous, 1x discrete)	no	built-in	⋆
	[65]	fairret	2024	dp, eod, eop, treq, other	3 (2x continuous, 1x discrete)	yes	built-in	⋆
post	[66]	RejectOptionClassification	2012	dp, eod, eop	1 (continuous)	no	implementation-based	●
	[67]	EqOddsPostprocesssing	2016	eod	-	no	implementation-based	●
	[68]	CalibratedEqOddsPostprocessing	2017	treq	-	no	implementation-based	●
	[69]	LevEqOpp	2019	eop	-	no	implementation-based	●
	[70]	JiangNachum	2020	dp, eod, eop	2 (1x continuous, 1x discrete)	yes	implementation-based	⋆
	[71]	DPAbstention	2021	dp	1 (continuous)	yes	-	-
	[72]	GetFair	2022	dp, eod, eop, treq	1 (continuous)	no	built-in	●
	[73]	FairBayesDPP	2022	other	1 (continuous)	no	-	●
	[74]	FaX	2022	consistency, other	-	no	built-in	●
	[75]	GroupDebias	2024	dp, eod	3 (1x continuous, 2x discrete)	no	-	⋆

TABLE I: List of fair binary classification approaches. Legend: dp = demographic parity; eod = equalized odds; eop = equal opportunity; treq = treatment equality; ● = data preparation optimization recommended, ⋆ = parameter optimization recommended. Highlighted rows indicate algorithms covered by previous experimental analysis papers.

A. Algorithm overview

We discuss pre-, in-, and post-processing algorithms, further divided into subcategories defined in a recent survey [14]. We ensure that we cover each subcategory with at least one competitive algorithm. Some algorithms fall into several subcategories as they combine different strategies. Our framework and evaluation focuses on approaches functioning *without* additional domain knowledge of the underlying data. Consequently, we neither cover causal fairness metrics nor consider causally fair classification approaches [16], [76].

1) Pre-processing: Relabelling and perturbation are methods that adjust the original labels or slightly modify the input data to remove or reduce bias. Representatives of relabeling attributes are *DisparateImpactRemover* [38], *LTDD* [43], *iFlipper* [44], *FairRR* [45], and *FairSSL* [41]. In the *FairSSL* [41] framework, biased instances are unlabeled and relabelled based on semi-supervised learning techniques, leading to four proposed variants labeled *Self Training* [77], *Co-Training* [78], *Label Propagation* [79], or *Label Spreading* [80]. We will combine the training strategies to *xT* and label strategies that are based on a graph structure to *Lx* and use the specific strategies of both categories as additional parameter.

Sampling techniques such as *Fair-SMOTE* [40] or *FS* [42] alter the distribution of the training data. They can involve oversampling underrepresented groups or undersampling overrepresented ones to achieve more balanced datasets.

Reweighing methods (e.g., [36]) retain the instances of the original dataset but assign different weights to them based on their labels and protected characteristics.

Representation learning aims to transform the input data into a representation that preserves the essential information for the predictions but removes the information related to the bias. This can be achieved by modifying the objective function to encourage the model to learn fair representations. Representatives of this category are *LFR* [37] and *PFR* [39].

2) In-processing: Regularization modifies the loss function by adding a fairness term to induce fairness during the classifiers' learning phase. Algorithms in this category: *PrejudiceRemover* [46], *HGR* [59], *FairDummies* [60], *HSICLinearRegression* [51], *ConvexFrameworkMode* [50], *SquaredDifferenceFairLogistic* [47], *FairGeneralizedLinearModel* [64], *GradualCompatibility* [63], and *fairret* [65].

	Metric	Definition
Correctness	Error rate	$\frac{FP+FN}{\lvert D\rvert}$
Group bias	Demographic parity (DP)	$\frac{1}{\lvert G\rvert-1}\sum_{i\in G}\left\lvert P(Z{=}1\lvert G_i) - P(Z{=}1)\right\rvert$
	Equalized odds (EOD)	$\frac{1}{\lvert G\rvert-1}\sum_{i\in G}\frac{1}{2}\left(\left\lvert FPR_{G_i} - FPR_{total}\right\rvert + \left\lvert TPR_{G_i} - TPR_{total}\right\rvert\right)$
	Treatment equality (TREQ)	$\frac{1}{\lvert G\rvert-1}\sum_{i\in G}\left\lvert \frac{FP_{G_i}}{FP_{G_j}+FN_{G_j}} - \frac{FP_{total}}{FP_{total}+FN_{total}}\right\rvert$
Individual bias	Consistency (CON)	$\frac{1}{\lvert D\rvert}\sum_{i\in D}\left\lvert Z_i - \frac{1}{k}\sum_{j\in \text{kNN}(X_i)} Z_j\right\rvert$

TABLE II: Metrics

Notation	Description
D	The whole dataset
G	A set of protected groups
X	Set of non-protected attributes
Y	Attribute denoting the ground-truth class label
Z	Attribute denoting the predicted class label
G_i	The protected group G for tuple $i \in D$
X_i	The non-protected attributes for tuple $i \in D$
Y_i	The ground truth Y for tuple $i \in D$
Z_i	The predicted class Z for tuple $i \in D$
$FP(R)/TP(R)$	False Positives (Rate)/True Positives (Rate)
$FN(R)/TN(R)$	False Negatives (Rate)/True Negatives (Rate)

TABLE III: Notations

Constraint optimization techniques incorporate fairness constraints into the optimization problem the learning algorithm aims to solve. Within this category, we consider the algorithms in [54], namely *GridSearchReduction* and *ExponentiatedGradientReduction*, *MetaFairClassifier* [55], *DisparateMistreatmentModel* [49], *FairnessConstraintModel* [48], *GeneralFairERM* [61], and *GerryFairClassifier* [52].

Adversarial learning involves training classification models and their adversaries simultaneously. While the classification model is trained on accuracy, an adversary is used to determine if the classifier is fair. The adversary is then used to tune the model. *Adversarial Debiasing* [53], *Fair Adversarial Gradient Tree Boosting (FAGTB)* [58], *GerryFairClassifier* [52], and *MultiAdversarialDebiasing* [62] fall in this category.

Compositional methods train several models, forming a model ensemble. The single models may not be suited to fairly and accurately classify instances on the whole data, but as an ensemble, fairness is achieved. This approach underlies *AdaFair* [56] and its extension [57], as well as *FAGTB* [58].

3) Post-processing: Input adjustments such as the approach by Jiang and Nachum [70] and *GroupDebias* [75], modify the input data. The difference from the pre-processing approaches is that the previous sample run's information is used to adapt the input data targeted for a specific classifier.

Output adjustments alter the output of the classifiers by adjusting thresholds with respect to probability distributions underlying binary classification tasks. Representatives of this category include *RejectOptionClassification* [66], *LevEqOpp* [69], and *FairBayesDPP* [73].

Classifier adjustment methods modify the predictive model after its training to correct for any bias. This can be achieved, e.g., by calibrating the classifier's predictions or learning a new transformation of the outputs that reduces the bias. We consider *EqOddsPostprocessing* [67], *CalibratedEqOddsPostprocessing* [68], *DPAbstention* [71], *FaX* [74], and *GetFair* [72].

B. Characteristics

Tab. I summarizes several algorithm characteristics relevant for our evaluation. The characteristics in the table refer to the publicly available implementation of these approaches.

Metric. Indicates for which fairness metric algorithms are designed. Most algorithms support demographic parity, equalized odds, and/or equal opportunity. Less approaches consider treatment equality or consistency (for individual fairness). Some further support other metrics, e.g., equalized expected log-likelihoods [64] or predictive parity [73].

Tuned parameters. While some of the pre- and post-processing approaches do not need any parameters (labeled $-$), all in-processing algorithms have at least one parameter that is tuned (see number). To keep aligned with the original work, we tune parameters also considered in the experiment section of the original papers. We further tune some available parameters commonly tuned in related approaches, i.e., the continuous fairness weight λ, a threshold for allowed fairness constraint violation ϵ, and the learning rate η. Some algorithms have additional parameters for which we use the default values.

Ignore protected. One pipeline component can be switched on to ignore protected attributes. Tab. I identifies the algorithms that support this component. Many have this component built-in, meaning that the protected attributes are automatically ignored during the training phase and only kept as metadata to optimize the models. Thus, this component does not affect these algorithms. *GridSearchReduction* [54] and *ExponentiatedGradientReduction* [54] provide this as a parameter option, supporting our component's functionality "out of the box". For the algorithms labeled as implementation-based, we were able to extend their implementations to support this option.

Multiple. This column relates to binarization, also part of data preparation in our pipeline (see Sec. II). Few approaches natively handle multiple protected attributes (labeled as yes). Some papers claim to allow or describe how to possibly extend to multiple protected attributes, but this remains theoretical by neither being implemented nor evaluated.

IV. Evaluation Metrics

Tab. II defines the metrics we use, relying on the notation in Tab. III. To measure the correctness of a classifier, we use the error rate (complement of accuracy). To assess fairness, we use bias metrics relating to either group fairness or individual fairness. For *group fairness* [11], [12], protected groups should have equal probabilities of an outcome. Thus, e.g., the probability of having a favorable outcome in a binary classification scenario should be independent of the protected attributes. The broad concept of global fairness incorporates various specific definitions, such as *demographic parity* [38], *equalized odds* [67], or *treatment equality* [81]. *Individual fairness* aims

dataset	sensitive attr.	# of samples	# of features	Pr(Y=1|G=1)	Pr(Y=1|G=0)
ACS2017 [85]	race	$72k$	23	49.6%	28.2%
ACS-ID [86]	sex	$8.2k$	10	37.7%	18.3%
ACS-OR [86]	sex	$22k$	10	42.7%	29.6%
Adult Data [87]	race	$46k$	21	26.3%	16%
Adult Data [87]	sex	$46k$	21	31.3%	11.4%
Adult Data [87]	race, sex	$46k$	21	32.4%	7.6% – 22.6%
Communities [88]	race	$2k$	91	62.6%	19.4%
COMPAS [89]	race	$6.1k$	7	50.2%	38.5%
COMPAS [89]	race, sex	$6.1k$	7	53%	33.6% – 39.9%
Credit Card [90]	sex	$30k$	23	24.2%	20.8%
German [91]	sex	$1k$	21	78.8%	75.5%
Implicit	sensitive	$15k$	8	65%	35%
Social	sensitive	$15k$	8	65%	35%

TABLE IV: Summary of benchmark datasets

for similar individuals being treated similarly [12]. This needs further specification of what similar means. One prominent individual fairness metric is *consistency* [37]. It calculates the average difference of the prediction of a person, compared to the average prediction of its k-nearest neighbors [82].

Our choices of bias metrics stem from their frequent use and because, unlike further metrics, they have not been shown to relate to one another. The range of all metrics is $[0, 1]$ - the lower, the better. For more details on further metrics, see [11], [13], [16], [83] (fairness) and [84] (correctness).

V. Evaluation

We conduct experiments with the overarching goal of gaining new insights into the performance of fairness-inducing algorithms in a broad pipeline context. This paper presents results for the following detailed research questions. We present further results in an extended version of this paper [21].

- **RQ1: synergies of data preparation and/or parameter optimization.** What is the effect of data preparation components and parameter optimization, in isolation or in combination, on the relative performance of algorithms, compared to default settings of isolated algorithms?
- **RQ2: combined effect of pipelined fairness-inducing algorithms.** Does the combination of several bias mitigation algorithms, i.e., a pre-processing algorithm to adapt the dataset combined with an in-/post-processing algorithm, improve individual approaches' results?
- **RQ3: cumulative improvements over time.** What progress can be attributed to the algorithmic contributions of the research community over more than a decade in terms of improving the accuracy-bias compromise?
- **RQ4: runtime and memory cost of pipeline configurations.** How do the algorithms scale in terms of runtime and memory usage in different pipeline configurations, and how may scalability concerns affect pipeline choices?

A. Setup

1) Datasets: We rely on real-world and synthetic datasets, summarized in Tab. IV, including probabilities of group associations. As real-world data, we use several datasets commonly used to benchmark fairness-aware ML approaches [92]. We further use two generated synthetic datasets. Each exhibits one of two biases: *Social* bias (aka direct bias) is the bias resulting solely from the sensitive attribute. For *implicit* bias, the sensitive attribute itself has no direct influence on the overall prediction but correlates with several of the other features that do. The selected synthetic datasets exhibit a substantial difference in probabilities (set to differ by 30% points) for the two groups, an underrepresented case in available real-world datasets. We further experimented with generated datasets with varying degrees of bias, observing that more extremely biased settings are more challenging to rectify (more and more algorithms struggle upholding fairness) and implicit bias is more difficult to compensate than social bias. Due to space constraints, details on this are only available in the extended version. We verified that the results reported in this section hold, whether or not the artificial data are considered. Since most algorithms do not apply to non-binary sensitive groups, we generate binary sensitive groups for the experiments, except for the binarization experiment (Sec.V-B1). All datasets are randomly split into 70% training and 30% validation data.

2) Algorithms: We compare algorithms introduced in Sec. III. We use the implementations provided by the AIF360 framework [23] for the available algorithms [36]–[38], [46], [52]–[55], [66]–[68]. Do et al. [93] provide an extensive set of algorithms [47], [49]–[51], [61], [64] which we reuse. For the remaining algorithms, we integrate the implementations in their available GitHub repositories into our framework.

3) Evaluation metric: A good fairness approach offers a good trade-off between fairness and accuracy. Thus, we evaluate the overall performance using the following score function for the prediction Z of a model over test data D:

$$score(D, Z) = 1 - (0.5 \cdot \text{error rate} + 0.5 \cdot \text{bias})$$

The bias term can use any of the metrics introduced in Sec. IV for a specific run. The score metric is also used as optimization function for the data preparation and hyperparameter optimization. Higher score values indicate better results. We also consider performance in terms of error rate and bias separately.

B. Data preparation and parameter optimization (RQ1)

We first study the impact of components preceding fairness-inducing algorithms on the overall performance, to determine if or to what extent different configurations and implementations of these components have a synergy effect. Analyzing the best performing pipeline implementations in various settings may allow us to find out if we can generally recommend certain configurations for certain algorithms.

We divide our discussion as follows: first, we investigate the effect the implemented fairness-related data preparation steps have. We then shift our attention to the remaining data preparation steps. We continue with a discussion of the effect of parameter optimization, before we conduct an end-to-end analysis involving all optimally configured components. As baseline, we use the default "out of the box" configurations of fairness-inducing algorithms. Throughout this section, we consider fairness-inducing algorithms one by one.

1) Fairness-related data preparation steps: This section focuses on the effect of different configuration options for the data preparation steps tied to fairness, i.e., ignoring protected

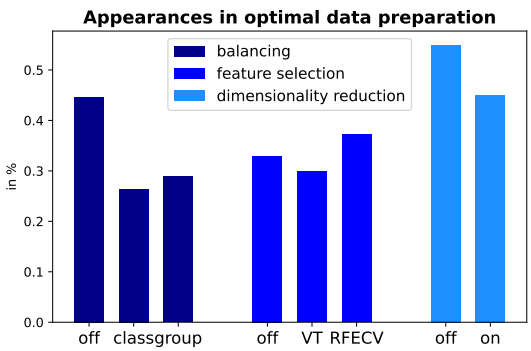


Fig. 2: Bars highlighting the appearance of each data preparation component in the optimal solution across all experiments.

attributes and binarization. The other preparation steps and parameter optimization are turned off.

According to Tab. I and our discussion in Sec. III, not all algorithms are amenable to the fairness-tied data preparation steps. Thus, the experiments are limited to the algorithms listed in Tab. I that allow some variation in these data preparation steps (i.e., 10 algorithms for binarization, 15 algorithms for ignoring the protected attributes) . Concerning the datasets, the binarization experiments are further limited to the *COMPAS* and *Adult Data Set*, as these are the only datasets with four protected groups to which binarization applies. For the evaluation, we ran the bias metrics over all groups. We run all possible experiments, i.e., whenever applicable, we consider both available options (on or off) for the binarization and protected attribute removal steps. This leaves us with two or four configurations per algorithm, dataset, and metric.

Tab. V summarizes the result of our analysis of the performance data across all these experiments. It **provides general fair data preparation configuration recommendations for each algorithm** based on the configurations performing among the best across at least 80% of experiments. A result is classified among the top, if its score is within at most 2% of the optimal solution. For instance, for *PFR* [39], for which both binarization and removal of protected attributes can be switched on or off, we observe that, in general, the binarization does not make a significant difference, while the removal component should be switched on.

Beyond the guidelines for individual algorithms, we observe that **ignoring protected attributes during the training phase rarely boosts the quality of fairness algorithms** (only in $\frac{4}{15}$ of algorithms). In the majority of cases, it either has no noticable effect or degrades the overall performance.

As highlighted in the introduction, it is commonly perceived as desirable that fair algorithms should be able to cover multiple protected groups simultaneously. However, in our set of experiments, the **binarization** of such groups into a single protected and single unprotected group **does not seem to have a negative effect on most algorithms** that are designed to handle multiple groups at once. Indeed, only 3 out of 10 algorithms (namely, *FairSSL-xT* [41], *FairSSL-Lx* [41] and *fairret* [65]) benefit from the direct support of multiple protected groups. The remaining algorithms, albeit supporting it as well, either perform equally well or better when binarization is performed.

Insight 1. *While ignoring protected attributes during the training phase and direct support for multiple protected groups are often assumed to be desirable, our experiments reveal that a majority of algorithms amenable to performing these or not are either insensitive or perform better when opting for the "unintuitive" choices. This first result highlights the importance of the end-to-end evaluation this paper conducts.*

This insight allows practitioners to fully benefit from the diversity of existing fairness-inducing algorithms, as there is no apparent need to favor any due to support of fairness-related data preparation or lack thereof.

2) General data preparation steps: We now turn our attention to the data preparation steps of balancing, feature selection, and dimensionality reduction. Each preparation step can be combined with each algorithm, resulting in 18 configurations per algorithm, dataset, and metric.

We discuss results when parameter optimization is turned off. For the fairness-related data preparation steps, we report results that follow our recommendations by activating the component of ignoring protected attributes only for the four algorithms generally benefiting from it (cf. Tab. V). Binarization is not necessary, as we report results only for datasets with one sensitive attribute to avoid algorithm penalties linked to (lack of) support of multiple sensitive attributes.

For each data preparation component implementation, **Fig. 2** reports how often it is part of the optimal data preparation setting (across all metrics, datasets, and model). We observe that **no variant for feature selection and dimensionality reduction stands out compared to the others**, preventing us from making a general recommendation. **For balancing, class balancing only rarely proves advantageous.**

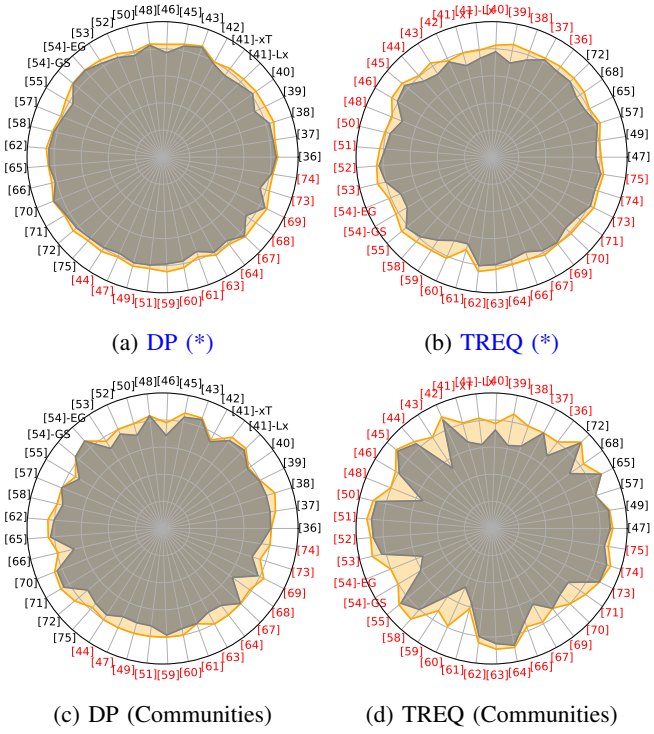
Drilling down on these results, we see a similar distribution for the *balancing* and *dimensionality reduction* components for all metrics. On the feature selection component, we recognize a deviation across the different metrics. For *demographic parity* and *equalized odds*, either *RFECV* as feature selection technique, or having the component turned off, appear in more optimal preparation components. When optimizing towards *consistency*, the difference is marginal. For *treatment equality*, there is a noticeable advantage gained from using one of the two feature selection strategies in most experiments(*VT* appears in 38.5% of the top results, *RFECV* in 34.7%).

Insight 2. *The overall effect and optimal choice of data preparation steps wrt end-to-end effectiveness varies depending on the targeted fairness metric, underlying dataset, and fairness-inducing algorithm. This renders optimally setting the data preparation challenging.*

Based on this insight, we further study pipeline implemen-

	[36]	[37]	[39]	[40]	[41]-xT	[41]-Lx	[45]	[52]	[54]-EG	[54]-GS	[58]	[65]	[66]	[67]	[68]	[69]	[70]	[71]
Binarization	×	×	-	-	■	■	×	-	-	●	×	■	×	×	×	×	-*	●
Ignoring prot.	■	-	●	-*	●	■	■	×	-	-	■	×	-*	●	●	■	-*	×

TABLE V: Fair data preparation configuration recommendations. Legend: ■: Deactivate, ●: Activate, -: Free choice, *: High uncertainty as no configuration combination is among the top results in at least 80% of experiments, ×: Activation not possible



(a) DP (*) (b) TREQ (*)

(c) DP (Communities) (d) TREQ (Communities)

Fig. 3: Optimal data preparation (orange) vs default (grey) for different metrics (DP and TREQ). No parameter tuning.

tations where data preparation optimization is performed in order to choose the best component implementations (i.e., maximizing *score*). We assume the optimization is done in a brute-force manner (i.e., running the 18 combinations and then choosing the best one). This procedure is comparable to the grid search employed in the parameter optimization component (relevant for our scalability discussion for RQ4).

Fig. 3(a–b) depict the overall average results when choosing the optimal data preparation configuration (orange) compared to the baseline of not performing any data preparation steps (grey). Due to space constraints, we limit to results on the DP and TREQ metrics, chosen as representative of the "average" case and the worst case, respectively. The references marked red denote algorithms that originally are not designed for the corresponding metric. In this plot, the *score* function ranges from 0 (center) to 1 (outer circle) with steps of 0.2 for each circle. We further showcase the detailed results on the *Communities* dataset in **Fig. 3(c–d)**, as this dataset show the biggest fluctuations. This is due to the underlying discrepancy within the dataset (cf. Tab. IV).

The results highlight the **performance-boost** when considering different data preparation techniques and optimizing the overall preparation pipeline towards the goal. Indeed, even when the improvements may seem rather small (e.g., for DP and *PrejudiceRemover* [46]), the detailed numbers reveal noticeable improvements (e.g., close to 6%-points). Of course, the scale of improvement varies depending on the specific metrics and datasets. For instance, on the *Communities* dataset, *PrejudiceRemover* improves on the baseline by approximately 10%-points when DP is the underlying fairness metric.

As *score* combines both accuracy and a fairness metric, we investigate to which of these individual performance indicators the overall improvement can be attributed. One key observation is that **fairness more frequently benefits from data preparation optimization than accuracy** (i.e., 76% vs 44% of all experiments). Furthermore, in the 40% of experiments where accuracy was reduced compared to the baseline, this **accuracy reduction was clearly outweighed by fairness improvement** (hence the overall positive effect wrt *score*).

> **Insight 3.** *Performing data preparation optimization can significantly increase the overall quality, in particular by improving fairness without jeopardizing accuracy.*

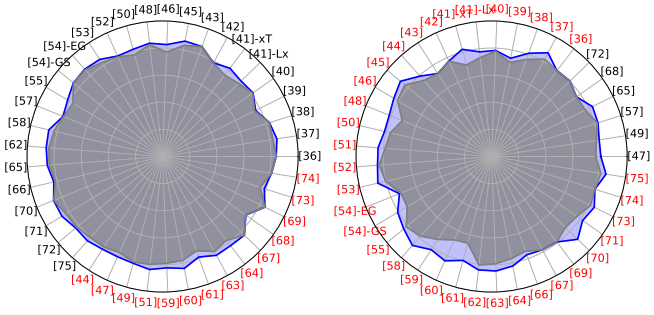
While this result is not very surprising, it raises the question if and how it combines or compares to the expected result of improved performance when applying parameter optimization.

3) Impact of parameter tuning: We first study its individual effect, deactivating all data preparation steps.

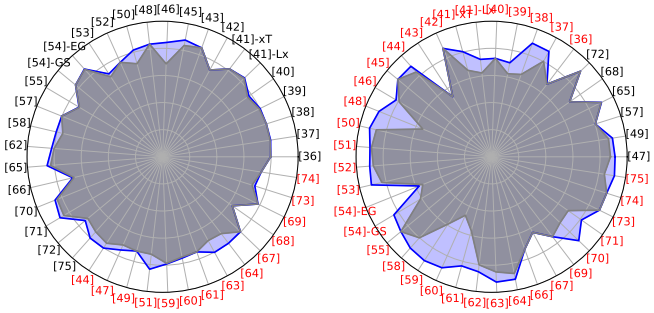
Analogously to Fig. 3, **Fig. 4(a–b)** depict the overall average results when performing parameter optimization (blue) compared to using the default parameter settings (grey). Detailed results on the *Communities* dataset are provided in **Fig. 4(c–d)**, where most fluctuations occur. Due to space constraints, we again limit to graphs for DT- and TREQ-based *score* metrics as average and worst-cacse representatives.

Overall, the results showcase a similar range of improvement to those achieved by data preparation optimization. That is, **the performance of algorithms can be significantly improved beyond their default configurations when applying parameter optimization**. Nevertheless, the better optimization (data preparation or parameter optimization) varies across experiments. For instance, the average improvement of *PrejudiceRemover* [46] when applying parameter optimization for DP amounts to 4.5%-points, as opposed to 6%-points for data preparation optimization. On the contrary, when focusing on the *Communities* dataset, parameter optimization obtains a significantly higher improvement over the baseline than data preparation optimization (16%-points vs 10%-points).

Anologously to our results on data preparation optimization, **fairness benefits more from parameter optimization than accuracy, but the difference is less pronounced** (i.e. 64% vs 50% of all experiments for algorithms with tuneable parameters).

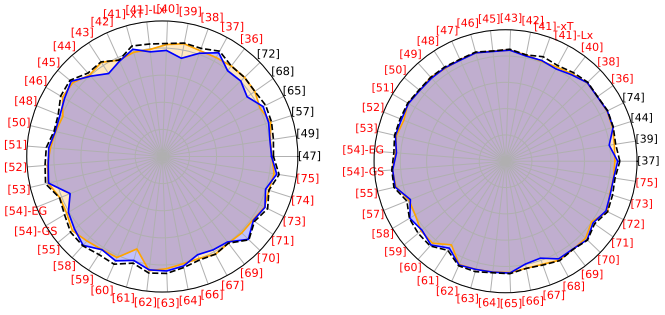


(a) DP (*) (b) TREQ (*)

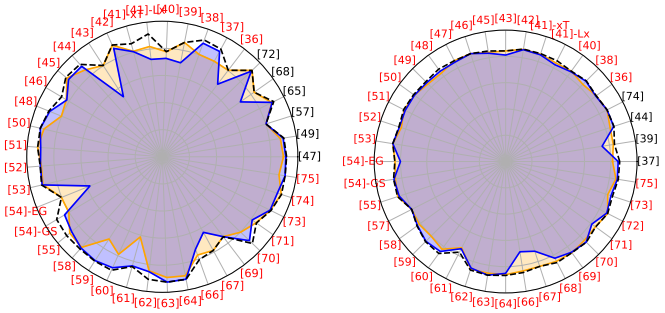


(c) DP (Communities) (d) TREQ (Communities)

Fig. 4: Parameter optimization (blue) vs default (grey) for DP and TREQ metrics. Data preparation components muted.



(a) TREQ (*) (b) CON (*)



(c) TREQ (Communities) (d) CON (Communities)

Fig. 5: Standalone optimal data preparation (orange) vs standalone optimal parameters (blue) vs combined optimal solution (dashed line).

Insight 4. *Parameter optimization, by itself, achieves a similar range of performance improvement over baseline executions as data preparation optimization. However, the distributions of improvements achieved by the two optimization strategies differ.*

This result leads to the follow-up question of how the results of both optimization variants compare to each other in detail and if, when combined, we can observe significant synergy effects through complementary individual behavior.

4) End-to-end performance analysis: This section analyzes performance when either data preparation optimization, parameter optimization, or both optimization strategies are "activated" in our pipeline.

Reusing the same visualization as before, we show the average results over all datasets in **Fig. 5(a–b)**, while **Fig. 5(c–d)** depict the results on the *Communities* dataset. This time, we depict the results when either using TREQ or CON as bias-metric within *score*. For TREQ, this presents an overlay of the previously seen colored areas (orange, blue, grey) for individually applying data preparation, parameter optimization, or the baseline, respectively. For CON, this implicitly reports additional results to previous experiments. The dashed black line depicts the performance attained when combining both optimization variants. For the six algorithms without tunable parameters (cf. Tab. I), the results for parameter optimization coincide with the results of the default configuration.

Looking at the performance of experiments for specific algorithms, we observe that for the majority of algorithms, the performance of parameter optimization and data preparation optimization are comparable. For instance, defining comparable optimal results as being within 2% (as explained before) of each other, *fairret* [65] results for both optimization variants are comparable in 69% of experiments. In the remaining experiments, parameter optimization is better than data preparation optimization (27% vs 3%). Thus, we recommend parameter optimization as the better default choice for fairret. Analogously, we can identify 9 additional algorithms for which parameter optimization is advisable. Similarly, we can recommend data preparation as primary optimization strategy for 14 algorithms (+ 6 algorithms where this is the only optimization option as they do not have tunable parameters). Tab. I shows these recommendations. Overall, **for 30 algorithms, we can recommend a default optimization strategy (either data preparation optimization or parameter optimization) that typically works well in the vast majority of settings.**

Irrespective of which optimization strategy performs better in isolation, the results show that **a combined optimization of data preparation and parameters does not significantly improve the results**, with a few exceptions visible on the *Communities* data. Due to the additional computational workload for joint optimizing, this is generally not recommended.

Insight 5. *Either using automatically optimized parameter settings (when an algorithm allows it) or optimally selected data preparation steps is both advised and sufficient when aiming for fair and accurate classifications.*

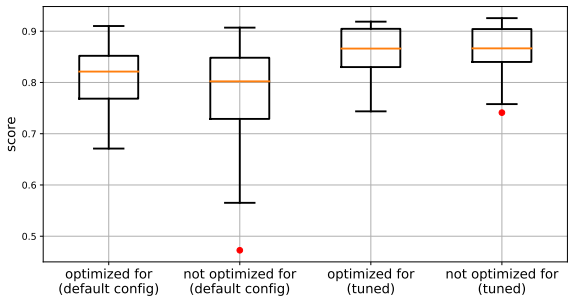


Fig. 6: Score distribution over all experiments on the *Communities* dataset

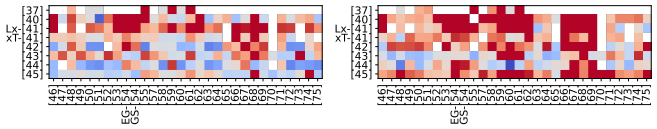
Conforming to results from earlier experimental studies [16], for the default configurations (grey), we observe throughout Fig. 3 – 4 that there is a trend that algorithms designed for a specific metric (black labels) perform (as expected) better than algorithms not designed for that metric (red labels). However, we observe that **data preparation or parameter optimization levels the field between algorithms specialized to one metric and non-specialized algorithms.** Additionally, optimization brings the "area shape" of the plots closer to a circle, making **most algorithms comparably suited to be used in a pipeline implementing one of the two optimization options.** This potentially simplifies the choice of a specific algorithm to be implemented as part of a pipeline, as no choice is a bad choice.

To further underline this, **Fig. 6** shows a boxplot of the scores across all metrics on the *Communities* dataset, both for the default configuration and the optimal setting (i.e., data preparation + parameter optimization). On this dataset, differences across algorithms are most pronounced and the score is typically lower than in other settings. The class labeled *Optimized for* groups algorithms that are specialized for the respective fairness metric, while *not optimized for* groups all other approaches. Clearly, after optimization, the score distribution looks nearly identical for both groups, while the default settings exhibit a clear difference.

Insight 6. *Optimal configurations reduce the potential impact on performance entailed by the choice of algorithm when targeting specific fairness metrics. Many algorithms reach comparable performance profiles, irrespective of their specialization to a particular metric.*

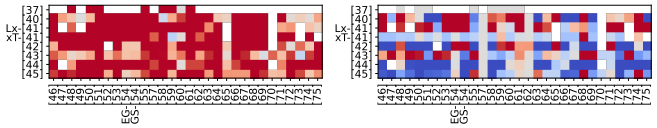
C. Combining fairness algorithms (RQ2)

Next, we study what happens when we combine fairness-inducing algorithms from different processing stages. For this set of experiments, we combine several algorithms of the pre-processing phase (LFR, Fair-SMOTE, FairSSL-Lx, FairSSL-xT, FS, LTDD, iFlipper and FairRR) with the other algorithms for the in-processing and post-processing stages. To further optimize the results, we apply parameter tuning. To obtain optimal parameters for algorithm combinations would significantly increase the grid size and the computational cost. Thus, we optimize the parameters consecutively: we first optimize the pre-processing algorithm, and then we use the optimized dataset output as input for the in-processing/post-processing algorithms, on which we perform parameter optimization again. We do not apply additional data preparation.



(a) DP (Communities) (b) EOD (Communities)

Fig. 7: Average *score* difference in % of each combination compared to the better of its baseline scores. Dark blue: 10% or more (improvement), dark red: -10% or less (degradation).



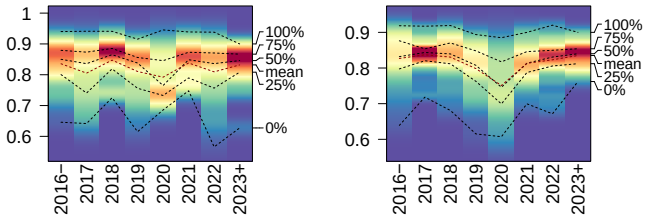
(a) accuracy (DP, Communities) (b) fairness (DP, Communities)

Fig. 8: Details on *score* difference: changes in accuracy and fairness component of *score*-function (encoding as in Fig. 7)

Fig. 7 depicts representative results on the *Communities* dataset when we combine a pre-processing algorithm (rows) with another in- or post-processing algorithm (columns) and optimize towards the *score*-function including different fairness metrics (i.e., (a) DP and (b) EOD). The overall trend and takeaway are similar for the experiments on the other datasets and metrics, where the extent (for both improvements and degradations) is less pronounced, though. The heatmaps encode the difference of *score* reached by an algorithm combination and the better of the two underlying algorithms. Blue cells represent improvements in performance, red cells degradation. White cells indicate missing values caused by errors thrown as some pre-processing algorithms alter the data such that it is no longer usable for the subsequent algorithm.

As the results show, **combining fairness approaches** from different processing stages **rarely improves the overall result** score over their baselines. There are some exceptions to this, e.g., for DP, there is an improvement for the combinations of *FS* [42] with either *GetFair* [72] or *FairBayesDPP* [73]. Such exceptions typically arise when the individual approaches themselves perform sub-optimally, and such combinations are either outperformed or at best marginally better than other (parameter-optimzied) individual algorithms.For instance, on the *Communities* dataset, the maximum improvement of the best scoring combination (*FairRR* [45]/*JiangNachum* [70]) over the corresponding best scoring single model (*fairret* [65]) is only 0.05%-points on the EOD metric.

To better understand the observed behavior, we study how the two components of the *score*-metric that balances accuracy and fairness vary between baselines and combinations. Reusing the same visualization and encoding as before, **Fig. 8** shows the differences in terms of (a) accuracy and (b) fairness



(a) EOD (b) CON

Fig. 9: Score distribution over all datasets. The algorithms are grouped by the publishing year.

for experiments optimizing *score* with DP as bias metric. Again, the results of the combined algorithm are compared to the better of the two individual algorithms, which could be a different baseline for accuracy and fairness. With this, we deliberately show the minimum performance difference, which still reveals significant differences. Our detailed analysis reveals that **combining fairness approaches** actually can **improve fairness** and that the overall score degradation typically stems from a huge **loss in accuracy**.

To explain this behavior, we go back to how pre-processing algorithms work. For *iFlipper* [44] and *FairRR* [45], only the labels are flipped. Meanwhile, *LFR* [37] and *LTDD* [43] alter the overall dataset (excluding the protected attributes). Hereby, the labels remain unchanged when applying *LTDD*, while *LFR* additionally adjusts its values. To achieve fairness, these algorithms actually alter the training data to such a significant extent (e.g., up to 50% labels flipped on the *Communities* dataset) that accurate predictions become difficult. As sampling is a component of the remaining pre-processing algorithms, a direct comparison to the original data is more difficult. In general, significant changes to the data made by pre-processing algorithms appear to be the main reason why accuracy scores plummet in combinations where subsequent algorithms operate on (too) significantly altered data.

Insight 7. *Combining fairness-inducing algorithms from different stages is rarely advisable. While an improvement of fairness can be achieved, the degradation in accuracy is generally bigger than the improvement of fairness.*

D. Improvement over time (RQ3)

Given that fairness-inducing algorithms have been studied for more than a decade, one could assume that new approaches have been proposed on the shoulders of previous ones to attain better performance (in specific aspects). We study how the overall performance has evolved over the years to answer the question if or how much the research community has advanced in devising algorithms that are (ideally) both fair and accurate.

Fig. 9 shows representative density heatmaps of the scoring results on the experiments using default settings over all datasets for both EOD and CON underlying the *score*-function. The algorithms are grouped by the year they have been published. We further combine all approaches from 2012-2016, as well as approaches from 2023 and 2024, ensuring a minimum number of 4 algorithms per bucket. The density is computed in relation to the respective bucket size. The plots further feature trend lines (e.g., mean, median at 50%, maximum at 100%, minimum at 0%).

These **results show that, overall, the accuracy-fairness-tradeoff has been stagnating** for binary classification tasks. This does not only hold true for the average results of the experiments, but also for the best solutions, which do not seem to be affected by the year the corresponding fairness approach has been published. Considering experiments with the fully-tuned pipeline (data preparation and parameter optimization) show the same trend, but less fluctuations.

Insight 8. *Current research on binary classification is stagnating regarding the accuracy-fairness tradeoff.*

This sparks several questions. For one, **do the benchmark datasets frequently used in the fairness community not allow much room for improvement?** Maybe new benchmark datasets are needed to evaluate the currently hidden potential of today's algorithms in more distinguishable and controllable settings. Secondly, **did we already peak with approaches from the past for the specific problem of binary classification?** Maybe it is time to switch focus on other fairness-related problems, e.g., to multi-class classification problems.

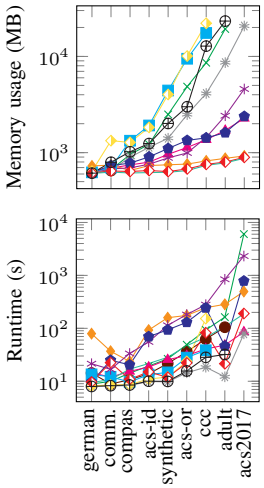
Some of the evaluated algorithms actually already expand the scope of applications: *FairGeneralizedLinearModel* [64] and *MultiAdversarialDebiasing* [62] are designed to further support multi-class classification problems. *MultiAdversarialDebiasing* further allows the definition of a single continuous protected attribute. Multiple approaches also support fair regression. However, the main focus of research still revolves around solely tackling binary classification problems.

Causal fairness approaches further address the root of unfairness and can provide a better reasoning, which helps to decide if the causes of unfairness are justifiable or not. However, this requires the involvement of domain experts.

E. Scalability (RQ4)

Our last series of experiments studies the scalability both in terms of memory usage and runtime.

1) Scalability in default settings: **Tab. VI (left)** depicts the memory usage and runtime for the default settings on the different datasets. For readability, we limit the plots to algorithms that do not scale well. One takeaway of the experimental paper by Islam et a. [16] is that post-processing algorithms are more efficient and scalable (in terms of runtime) than pre-processing and in-processing algorithms. With the increased set of algorithms covered from each processing stage, this **statement can be refuted**. The results showcase that the post-processing algorithms *GetFair* [72] and *FairBayesDPP* [73] are among the least runtime-efficient algorithms. This also holds for memory usage, for which *FairBayesDPP* [73], *RejectOptionClassification* [66] and *DPAbstention* [71] do not scale well. We further observe that there are a few fairness approaches that run out of memory on larger datasets, while others are prohibitively expensive in terms of runtime.



		[36]	[37]	[38]	[39]	[40]	[41]-Lx	[41]-xT	[42]	[43]	[44]	[45]	[66]	[67]	[68]	[69]	[70]	[71]	[72]	[73]	[74]	[75]
	(a)	711	795	780	8607	792	847	811	750	714	5141	723	1363	719	726	731	786	17536	821	1439	736	743
(1)	(b)	711	**1258**	929	12073	**966**	1121	**1032**	784	714	7971	871	**3403**	719	726	731	**1041**	17606	**1033**	1464	736	921
	(c)	**939**	969	**1078**	**24785**	923	**1798**	1002	**1075**	**966**	**11350**	**978**	1935	**935**	**950**	**942**	1023	**25777**	960	**3065**	**951**	**937**
	(a)	9	63	14	62	43	23	22	15	11	93	10	41	10	10	15	16	39	245	280	10	11
(2)	(b)	9	**4242**	37	884	251	228	205	27	11	687	20	131	10	10	15	178	266	1509	1022	10	**551**
	(c)	**373**	1665	**476**	**2208**	**634**	**1147**	**819**	**591**	**370**	**5388**	**388**	**1143**	**404**	**402**	**455**	**544**	**1649**	**3413**	**13831**	**384**	377
(3)					●						×		▲					■	◆	✳		

		[46]	[47]	[48]	[49]	[50]	[51]	[52]	[53]	[54]-EG	[54]-GS	[55]	[57]	[58]	[59]	[60]	[61]	[62]	[63]	[64]	[65]
	(a)	871	4124	1058	1425	22112	706	748	743	733	712	777	732	756	812	738	927	720	717	12803	763
(1)	(b)	898	4195	1357	1548	25292	772	802	781	733	769	857	**1016**	**1205**	874	**992**	**1209**	891	**1460**	12921	**1007**
	(c)	**1086**	**12261**	**1407**	**2041**	**26183**	**950**	**936**	**942**	**912**	**926**	**980**	951	936	**961**	919	1181	**944**	911	**25432**	911
	(a)	44	18	20	243	153	10	12	22	16	16	12	41	82	58	68	14	10	14	28	34
(2)	(b)	176	103	110	546	2417	16	22	64	122	70	22	**2006**	2107	1077	**2532**	61	**452**	**7638**	332	953
	(c)	**1239**	**675**	**600**	**5096**	**6526**	**367**	**408**	**626**	**435**	**514**	**406**	1330	**2895**	**1538**	1877	**464**	375	521	**1161**	**1030**
(3)			✳		⬟	◆								◆						⊕	

TABLE VI: Efficiency experiments with the default configurations on different datasets for selected algorithms with worse scalability (left) and more detailed analysis on the credit card clients data (right): (1) maximum memory usage; (2) total runtime; (3) symbol for the figure on the left – (a) default configuration; (b) hyperparameter tuning; (c) data preparation optimization

Insight 9. *Scalability in terms of runtime and memory is independent of the processing stage of an algorithm. Yet, there are considerable differences in terms of runtime and memory scalability across algorithms.*

From a practical perspective, this means that different scalability requirements influence the choice of algorithm, possibly to a larger extent than accuracy or fairness (as most algorithms, when optimized, perform reasonably well in that respect).

2) Optimization overhead: The overhead introduced by either data preparation optimization or parameter optimization is illustrated in **Tab. VI (right)** for experiments on the *Credit Card* dataset. We chose this dataset as it is the largest one for which all algorithms run. The table reports (1) the maximum memory usage and (2) total runtime for (a) the default configuration, (b) parameter optimization, and (c) data preparation optimization. When the algorithms are featured in the scalability graphs (left of the table), the corresponding symbol is shown in the line labeled (3).

We see that across our experiments and grid spaces, **optimizing the data preparation configuration typically results in higher runtime and memory consumption**. We attribute this behavior to the balancing component, which mixes oversampling and undersampling, but typically increases the overall amount of tuples in the dataset. This entails, in addition to the time needed by the data preparation step itself, an increase of memory usage and runtime for the fairness-inducing algorithm. The other preparation components may reduce the overall memory usage due to reducing features. This can also result in the reduction of runtime, but may be countered by the time needed for the components to run. **Exceptions arise when the cost of parameter optimization is exacerbated through specific parameters**, e.g., directly affecting the required memory [57] or entailing a large grid size for parameter optimization (e.g., [37], [63]).

Insight 10. *Data preparation components do have an impact on memory usage and overall runtime. In our implementation, parameter optimization is more scalable than optimizing the data preparation configuration.*

We acknowledge that the time-consuming data preparation optimization could be improved using dedicated frameworks optimizing the preparation process (e.g., [94]–[96]).

VI. Conclusion

We conducted an extensive experimental study on the performance of fair classification algorithms within a multi-step pipeline comprising alternative implementations for each step.

Our experimental study revealed that practitioners are flexible in either optimizing data preparation or parameter optimization, without the necessity to consider both. By doing so, choosing a suited fairness-inducing algorithm for their application becomes easy, as most algorithms represent good choices that balance accuracy and fairness and there is no need to combine algorithms operating at different processing stages (pre-, in-, and post-processing). On the other hand, algorithm scalability varies significantly. Our evaluation provides guidance on which algorithms should rather be chosen when aiming for either runtime or memory scalability.

We further discovered that algorithms developed throughout the years do not exhibit the generally desired improvement of performance over previous algorithms. This raises interesting questions for future research, at methodological, algorithm focus, and benchmarking levels.

Acknowledgment

Supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2120/1 – 390831618 and an NTU SUG.

REFERENCES

[1] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 34, no. 11, pp. 2767–2787, 2010.

[2] L. Li, T. Lassiter, J. Oh, and M. K. Lee, "Algorithmic hiring in practice: Recruiter and hr professional's perspectives on ai use in hiring," in *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 2021, pp. 166–176.

[3] P. J. Bickel, E. A. Hammel, and J. W. O'Connell, "Sex bias in graduate admissions: Data from berkeley: Measuring bias is harder than is usually assumed, and the evidence is sometimes contrary to expectation." *Science*, vol. 187, no. 4175, pp. 398–404, 1975.

[4] T. Brennan, W. Dieterich, and B. Ehret, "Evaluating the predictive validity of the compas risk and needs assessment system," *Criminal Justice and behavior*, vol. 36, no. 1, pp. 21–40, 2009.

[5] J. Dastin, "Amazon scraps secret AI recruiting tool that showed bias against women," *Reuters.com*, 2018.

[6] J. Carpenter, "Google's algorithm shows prestigious job ads to men, but not to women. here's why that should worry you." *The Washington Post*, 2015.

[7] K. Lum and W. Isaac, "To predict and serve?" *Significance*, vol. 13, no. 5, pp. 14–19, 2016.

[8] A. Chhabra, K. Masalkovaitė, and P. Mohapatra, "An overview of fairness in clustering," *IEEE Access*, vol. 9, pp. 130 698–130 720, 2021.

[9] M. Zehlike, K. Yang, and J. Stoyanovich, "Fairness in ranking, part i: Score-based ranking," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–36, 2022.

[10] ——, "Fairness in ranking, part ii: Learning-to-rank and recommender systems," *ACM Computing Surveys*, vol. 55, no. 6, pp. 1–41, 2022.

[11] S. Caton and C. Haas, "Fairness in machine learning: A survey," *ACM Comput. Surv.*, vol. 56, no. 7, apr 2024. [Online]. Available: https://doi.org/10.1145/3616865

[12] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[13] D. Pessach and E. Shmueli, "A review on fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 55, no. 3, pp. 1–44, 2022.

[14] M. Hort, Z. Chen, J. M. Zhang, M. Harman, and F. Sarro, "Bias mitigation for machine learning classifiers: A comprehensive survey," *ACM J. Responsib. Comput.*, vol. 1, no. 2, jun 2024. [Online]. Available: https://doi.org/10.1145/3631326

[15] S. Hajian and J. Domingo-Ferrer, "Direct and indirect discrimination prevention methods," in *Discrimination and privacy in the information society*. Springer, 2013, pp. 241–254.

[16] M. T. Islam, A. Fariha, A. Meliou, and B. Salimi, "Through the data management lens: Experimental analysis and evaluation of fair classification," in *Proceedings of the 2022 International Conference on Management of Data*, 2022, pp. 232–246.

[17] J. R. Foulds, R. Islam, K. N. Keya, and S. Pan, "An intersectional definition of fairness," in *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 2020, pp. 1918–1921.

[18] T. Makkonen, "Multiple, compoud and intersectional discrimination: bringing the experiences of the most marginalized to the fore," 2002.

[19] M. A. Haeri and K. A. Zweig, "The crucial role of sensitive attributes in fair classification," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2020, pp. 2993–3002.

[20] T. Calders and S. Verwer, "Three naive bayes approaches for discrimination-free classification," *Data mining and knowledge discovery*, vol. 21, pp. 277–292, 2010.

[21] N. Lässig, "Fair classification pipeline experiments repository," 2024. [Online]. Available: https://github.com/NicoLaessig/fair-classification-pipeline-exps

[22] M. Hort, J. M. Zhang, F. Sarro, and M. Harman, "Fairea: A model behaviour mutation approach to benchmarking bias mitigation methods," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 994–1006.

[23] R. K. E. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilovic, S. Nagar, K. N. Ramamurthy, J. Richards, D. Saha, P. Sattigeri, M. Singh, K. R. Varshney, and Y. Zhang, "AI Fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias," Oct. 2018. [Online]. Available: https://arxiv.org/abs/1810.01943

[24] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth, "A comparative study of fairness-enhancing interventions in machine learning," in *Proceedings of the conference on fairness, accountability, and transparency*, 2019, pp. 329–338.

[25] Z. Chen, J. M. Zhang, F. Sarro, and M. Harman, "A comprehensive empirical study of bias mitigation methods for machine learning classifiers," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 4, pp. 1–30, 2023.

[26] S. Guha, F. A. Khan, J. Stoyanovich, and S. Schelter, "Automated data cleaning can hurt fairness in machine learning-based decision making," *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[27] S. Biswas and H. Rajan, "Fair preprocessing: towards understanding compositional fairness of data transformers in machine learning pipeline," in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 981–993.

[28] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, 2020.

[29] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization." *Journal of machine learning research*, vol. 13, no. 2, 2012.

[30] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. De Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2015.

[31] G. E. Batista, A. L. Bazzan, M. C. Monard *et al.*, "Balancing training data for automated annotation of keywords: a case study." *Wob*, vol. 3, pp. 10–8, 2003.

[32] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, "Feature selection: A data perspective," *ACM computing surveys (CSUR)*, vol. 50, no. 6, pp. 1–45, 2017.

[33] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine learning*, vol. 46, pp. 389–422, 2002.

[34] T. Minka, "Automatic choice of dimensionality for pca," *Advances in neural information processing systems*, vol. 13, 2000.

[35] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[36] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," *Knowledge and information systems*, vol. 33, no. 1, pp. 1–33, 2012.

[37] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, "Learning fair representations," in *International conference on machine learning*. PMLR, 2013, pp. 325–333.

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

[39] P. Lahoti, K. P. Gummadi, and G. Weikum, "Operationalizing individual fairness with pairwise fair representations," *Proc. VLDB Endow.*, vol. 13, no. 4, p. 506–518, dec 2019. [Online]. Available: https://doi.org/10.14778/3372716.3372723

[40] J. Chakraborty, S. Majumder, and T. Menzies, "Bias in machine learning software: Why? how? what to do?" in *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2021, pp. 429–440.

[41] J. Chakraborty, S. Majumder, and H. Tu, "Fair-ssl: Building fair ml software with less data," in *Proceedings of the 2nd International Workshop on Equitable Data and Technology*, 2022, pp. 1–8.

[42] T. Zhang, T. Zhu, J. Li, M. Han, W. Zhou, and P. S. Yu, "Fairness in semi-supervised learning: Unlabeled data help to reduce discrimination," *IEEE Transactions on Knowledge & Data Engineering*, vol. 34, no. 04, pp. 1763–1774, apr 2022.

[43] Y. Li, L. Meng, L. Chen, L. Yu, D. Wu, Y. Zhou, and B. Xu, "Training data debugging for the fairness of machine learning software," in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 2215–2227.

[44] H. Zhang, K. H. Tae, J. Park, X. Chu, and S. E. Whang, "iflipper: Label flipping for individual fairness," *Proceedings of the ACM on Management of Data*, vol. 1, no. 1, pp. 1–26, 2023.

[45] J. J. Ward, X. Zeng, and G. Cheng, "Fairrr: Pre-processing for group fairness through randomized response," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2024, pp. 3826–3834.

[46] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, "Fairness-aware classifier with prejudice remover regularizer," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23*. Springer, 2012, pp. 35–50.

[47] Y. Bechavod and K. Ligett, "Penalizing unfairness in binary classification," *arXiv preprint arXiv:1707.00044*, 2017.

[48] M. B. Zafar, I. Valera, M. G. Rogriguez, and K. P. Gummadi, "Fairness constraints: Mechanisms for fair classification," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 962–970.

[49] M. B. Zafar, I. Valera, M. Gomez Rodriguez, and K. P. Gummadi, "Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 1171–1180.

[50] R. Berk, H. Heidari, S. Jabbari, M. Joseph, M. Kearns, J. Morgenstern, S. Neel, and A. Roth, "A convex framework for fair regression," *arXiv preprint arXiv:1706.02409*, 2017.

[51] A. Pérez-Suay, V. Laparra, G. Mateo-García, J. Muñoz-Marí, L. Gómez-Chova, and G. Camps-Valls, "Fair kernel learning," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2017, Skopje, Macedonia, September 18–22, 2017, Proceedings, Part I*. Springer, 2017, pp. 339–355.

[52] M. Kearns, S. Neel, A. Roth, and Z. S. Wu, "Preventing fairness gerrymandering: Auditing and learning for subgroup fairness," in *International conference on machine learning*. PMLR, 2018, pp. 2564–2572.

[53] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating unwanted biases with adversarial learning," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 335–340.

[54] A. Agarwal, A. Beygelzimer, M. Dudík, J. Langford, and H. Wallach, "A reductions approach to fair classification," in *International Conference on Machine Learning*. PMLR, 2018, pp. 60–69.

[55] L. E. Celis, L. Huang, V. Keswani, and N. K. Vishnoi, "Classification with fairness constraints: A meta-algorithm with provable guarantees," in *Proceedings of the conference on fairness, accountability, and transparency*, 2019, pp. 319–328.

[56] V. Iosifidis and E. Ntoutsi, "Adafair: Cumulative fairness adaptive boosting," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 781–790.

[57] V. Iosifidis, A. Roy, and E. Ntoutsi, "Parity-based cumulative fairness-aware boosting," *Knowledge and Information Systems*, vol. 64, no. 10, pp. 2737–2770, 2022.

[58] V. Grari, B. Ruf, S. Lamprier, and M. Detyniecki, "Fair adversarial gradient tree boosting," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 1060–1065.

[59] J. Mary, C. Calauzenes, and N. El Karoui, "Fairness-aware learning for continuous attributes and treatments," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4382–4391.

[60] Y. Romano, S. Bates, and E. Candes, "Achieving equalized odds by resampling sensitive attributes," *Advances in neural information processing systems*, vol. 33, pp. 361–371, 2020.

[61] L. Oneto, M. Donini, and M. Pontil, "General fair empirical risk minimization," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[62] A. Ball-Burack, M. S. A. Lee, J. Cobbe, and J. Singh, "Differential tweetment: Mitigating racial dialect bias in harmful tweet detection," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 2021, pp. 116–128.

[63] C. Hertweck and T. Räz, "Gradual (in) compatibility of fairness criteria," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 11, 2022, pp. 11 926–11 934.

[64] H. Do, P. Putzel, A. S. Martin, P. Smyth, and J. Zhong, "Fair generalized linear models with a convex penalty," in *International Conference on Machine Learning*. PMLR, 2022, pp. 5286–5308.

[65] M. Buyl, M. Defrance, and T. D. Bie, "fairret: a framework for differentiable fairness regularization terms," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=NnyD0Rjx2B

[66] F. Kamiran, A. Karim, and X. Zhang, "Decision theory for discrimination-aware classification," in *2012 IEEE 12th international conference on data mining*. IEEE, 2012, pp. 924–929.

[67] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[68] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, and K. Q. Weinberger, "On fairness and calibration," *Advances in neural information processing systems*, vol. 30, 2017.

[69] E. Chzhen, C. Denis, M. Hebiri, L. Oneto, and M. Pontil, "Leveraging labeled and unlabeled data for consistent fair binary classification," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[70] H. Jiang and O. Nachum, "Identifying and correcting label bias in machine learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 702–712.

[71] N. Schreuder and E. Chzhen, "Classification with abstention but without disparities," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 1227–1236.

[72] S. Sikdar, F. Lemmerich, and M. Strohmaier, "Getfair: Generalized fairness tuning of classification models," in *2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022, pp. 289–299.

[73] X. Zeng, E. Dobriban, and G. Cheng, "Fair bayes-optimal classifiers under predictive parity," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 692–27 705, 2022.

[74] P. A. Grabowicz, N. Perello, and A. Mishra, "Marrying fairness and explainability in supervised learning," in *2022 ACM Conference on Fairness, Accountability, and Transparency*, 2022, pp. 1905–1916.

[75] E. Chan, Z. Liu, R. Qiu, Y. Zhang, R. Maciejewski, and H. Tong, "Group fairness via group consensus," in *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, 2024, pp. 1788–1808.

[76] J. Pearl, *Causality*. Cambridge university press, 2009.

[77] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," in *33rd annual meeting of the association for computational linguistics*, 1995, pp. 189–196.

[78] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*, 1998, pp. 92–100.

[79] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," *ProQuest Number: INFORMATION TO ALL USERS*, 2002.

[80] D. Zhou, O. Bousquet, T. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," *Advances in neural information processing systems*, vol. 16, 2003.

[81] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth, "Fairness in criminal justice risk assessments: The state of the art," *Sociological Methods & Research*, vol. 50, no. 1, pp. 3–44, 2021.

[82] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[83] I. Žliobaitė, "Measuring discrimination in algorithmic decision making," *Data Mining and Knowledge Discovery*, vol. 31, no. 4, pp. 1060–1089, 2017.

[84] Ž. Vujović *et al.*, "Classification model evaluation metrics," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 599–606, 2021.

[85] U. C. Bureau, "Us census demographic data," https://www.kaggle.com, 2019.

[86] F. Ding, M. Hardt, J. Miller, and L. Schmidt, "Retiring adult: New datasets for fair machine learning," *Advances in neural information processing systems*, vol. 34, pp. 6478–6490, 2021.

[87] D. Dua and C. Graff, "Adult data set. UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/adult

[88] ——, "Communities and crime data set. UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/communities+and+crime

[89] ProPublica, "Compas recidivism racial bias," https://www.kaggle.com, 2017.

[90] D. Dua and C. Graff, "default of credit card clients data set. UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

[91] ——, "Statlog (german credit data) data set. UCI machine learning repository," 2017. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)

[92] T. Le Quy, A. Roy, V. Iosifidis, W. Zhang, and E. Ntoutsi, "A survey on datasets for fairness-aware machine learning," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 12, no. 3, p. e1452, 2022.

[93] H. Do and P. Putzel, "Fair generalized linear models with a convex penalty," https://github.com/hyungrok-do/fair-glm-cvx, 2022.

[94] P. Li, Z. Chen, X. Chu, and K. Rong, "Diffprep: Differentiable data preprocessing pipeline search for learning over tabular data," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 1–26, 2023.
[95] G. Zhu, Z. Xu, C. Yuan, and Y. Huang, "Difer: differentiable automated feature engineering," in *International Conference on Automated Machine Learning*. PMLR, 2022, pp. 17–1.
[96] L. Berti-Equille, "Learn2clean: Optimizing the sequence of tasks for web data preparation," in *The world wide web conference*, 2019, pp. 2580–2586.

VII. Additional Experiments

We further conducted other experiments to measure the performance under different data distributions. For this, we varied the bias on our synthetic datasets.

A. Performance shift under varying bias distributions in the synthetic setting

For the experiments within this paper, we evaluate the algorithms on real-world data as well as two synthetic datasets. Each exhibits one of two biases: *Social* bias induces direct bias, while for the *implicit* bias datasets indirect bias was induced. We used the datasets that have a 30% difference in bias for the groups. For this set of experiments, we vary the bias between 10% and 50%. To better understand the influence on changing data distributions on the overall outcome, and how well both types of bias are mitigated, we now apply the algorithms to the synthetic datasets. The results on the implicit bias datasets are depicted in **Fig. 10** and the results on the social bias datasets are shown in **Fig. 11**. These density heatmaps display the score over all algorithms for the different metrics. Furthermore, due to previous results of this experimental paper, the data preparation optimization is activated, while the hyperparameter tuning component is muted.

As we can see, the **performance of fair classifications decrease with the increase of bias distribution within the dataset**. For both types of biases, the performance degradation is more noticeable with a bias of over 30%. Overall these results are expected. Since the baseline contains higher bias, achieving fairness requires a bigger tradeoff of accuracy. Analyzing the results in more detail, the error rate of the fair classifiers is quite constant over the different synthetic datasets and the performance degradation is mainly caused by the algorithms not being able to mitigating bias as well.

Upon further analysis, it becomes evident that **activating optimization components gains relevance as the bias gap of the original dataset increases**. While the error rate difference of the optimization revolves around 0% for the majority of the algorithms, the reduction of biased outcomes increases. Thus, the impact of bias distributions is even more prominent when using the default configuration.

> **Insight 11.** *Bias distributions affect the overall outcome quality and mainly affect the overall group bias of the results. While optimization components reduce this effect, it still persists.*

The results also show that the **classifiers typically tackle social bias better than implicit bias**. This is also expected, as social bias is directly caused by the protected attributes. Thus, identifying and resolving this type of bias is less complex.

> **Insight 12.** *The type of bias within the datasets influences the overall quality improvements of fairness-inducing approaches. The approaches typically tackle social bias better than implicit bias.*

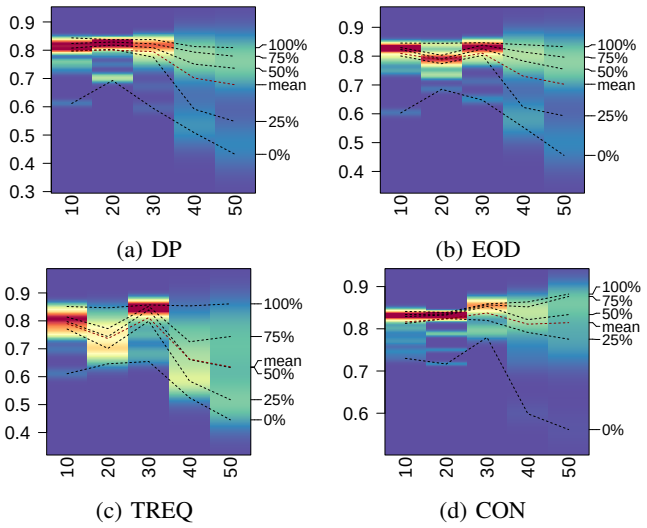


Fig. 10: Score distributions of the tuned results over different implicit bias distributions.

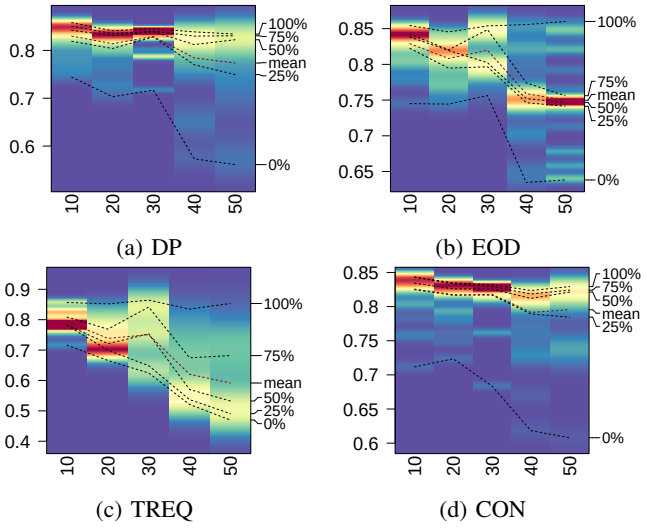


Fig. 11: Score distributions of the tuned results over different social bias distributions.