Fairkit-learn: A Fairness Evaluation and Comparison Toolkit

Brittany Johnson
George Mason University
Fairfax, VA, USA
johnsonb@gmu.edu

Yuriy Brun
University of Massachusetts Amherst
Amherst, MA, USA
brun@cs.umass.edu

ABSTRACT

Advances in how we build and use software, specifically the integration of machine learning for decision making, have led to widespread concern around model and software fairness. We present fairkit-learn, an interactive Python toolkit designed to support data scientists' ability to reason about and understand model fairness. We outline how fairkit-learn can support model training, evaluation, and comparison and describe the potential benefit that comes with using fairkit-learn in comparison to the state-of-the-art. Fairkit-learn is open source at https://go.gmu.edu/fairkit-learn/.

CCS CONCEPTS

• **Software and its engineering** → **Software testing and debugging**.

KEYWORDS

Software fairness, bias-free software design, visualization

ACM Reference Format:
Brittany Johnson and Yuriy Brun. 2022. Fairkit-learn: A Fairness Evaluation and Comparison Toolkit. In *44th International Conference on Software Engineering Companion (ICSE '22 Companion), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3510454.3516830

1 INTRODUCTION

Software engineering and data scientists, more and more, use data to train machine learning models as part of software systems. Not only is data-driven software becoming more pervasive, it is being adopted in contexts where unexpected outcomes can have detrimental impact. From who gets a job [29], to the diagnosis and treatment of medical patients [32], data-driven software affects many important decisions.

While there is potential for data-driven software to improve our way of life, recent studies suggest that societal biases are in the data we use when training machine learning models, which leads to technological biases [12, 27]. YouTube makes more mistakes when rendering closed captions for female voices [22, 33]. E-commerce software has showcased bias in their services and discounts [16, 25]. Facial recognition software has difficulty accurately recognizing female and non-white faces [6, 17, 21]. Unfortunately, the list goes on and on.

ICSE '22 Companion, May 21–29, 2022, Pittsburgh, PA, USA

ACM ISBN 978-1-4503-9223-5/22/05…$15.00
https://doi.org/10.1145/3510454.3516830

```python
models = {'LogisticRegression': LogisticRegression,
          'RandomForestClassifier': RandomForestClassifier,
          'AdversarialDebiasing': AdversarialDebiasing}

metrics = {'UnifiedMetricLibrary': [UnifiedMetricLibrary,
                                    'accuracy_score',
                                    'average_odds_difference',
                                    'statistical_parity_difference',
                                    'equal_opportunity_difference',
                                    'disparate_impact'
                                   ]

processor_args = {'unprivileged_groups': unprivileged, 'privileged_groups': privileged}
hyperparameters = {'LogisticRegression':{'penalty': ['l1', 'l2'], 'C': [0.1, 0.5, 1]},
                   'RandomForestClassifier':{},
                  'AdversarialDebiasing': DEFAULT_ADB_PARAMS(**processor_args)
                  }

thresholds = [i * 10.0/100 for i in range(5)]

preprocessors=[DisparateImpactRemover(), Reweighing(**processor_args)]
postprocessors=[CalibratedEqOddsPostprocessing(**processor_args)]
```

Figure 1: Example parameters for model search in fairkit-learn

One way to support data scientists in addressing bias in machine learning is by providing tools that can help them reason about the various considerations that come with training high quality, unbiased models. To this end, we developed fairkit-learn, an open source Python toolkit for evaluating and comparing machine learning models with respect to fairness and other quality metrics [20]. An evaluation of fairkit-learn found that it does in fact support the ability to find models that are both fair and high quality, and improves the ability to do so over scikit-learn and AI Fairness 360. This paper outlines the components of fairkit-learn and how it can be used to automatically (and with little overhead for the user) train, evaluate, and compare a large number of model configurations.

Next, Section 2 describes fairkit-learn and Section 3 details how it can be used to help develop fair software. Section 4 places our research in the context of related work, and Section 5 summarizes our contributions. A video of fairkit-learn in action is available at https://youtu.be/ZC_deJnI9xs/.

2 FAIRKIT-LEARN: MODEL EVALUATION AND COMPARISON

Figure 2 outlines the steps involved when using fairkit-learn. Along with the dataset being used for model training, the inputs to fairkit-learn are *models*, *hyperparameters*, *metrics*, *protected attribute*, *classification threshold*, and *pre- and post-processing algorithms*. In this section, we discuss the various components of fairkit-learn and how it uses these inputs.

Integrated machine learning tools. Fairkit-learn is built on top of scikit-learn and AIF360 [19, 30]. Given scikit-learn is a foundational machine learning toolkit, we wanted to make sure fairkit-learn could interface with its algorithms and metrics. We integrated AIF360, which also builds on top of scikit-learn, to provide fairkit-learn's fairness-related functionality.

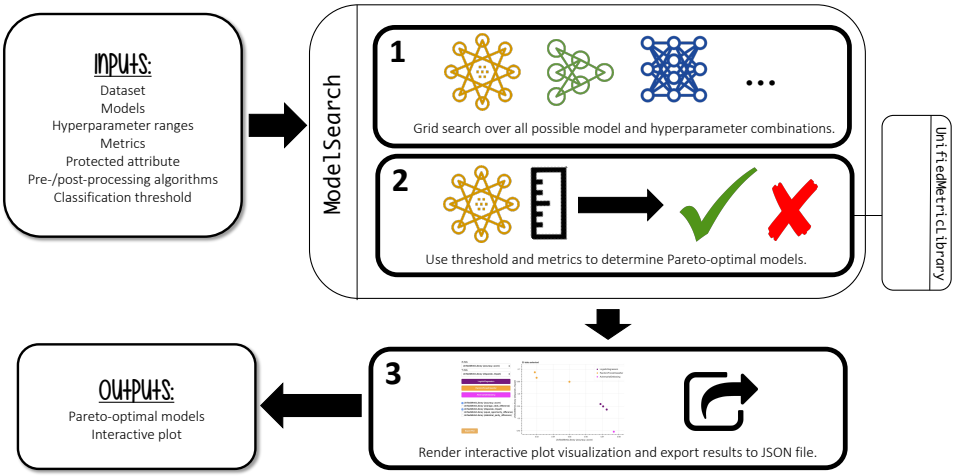

Figure 2: Fairkit-learn workflow, all of which takes place within your Python code and execution environment.

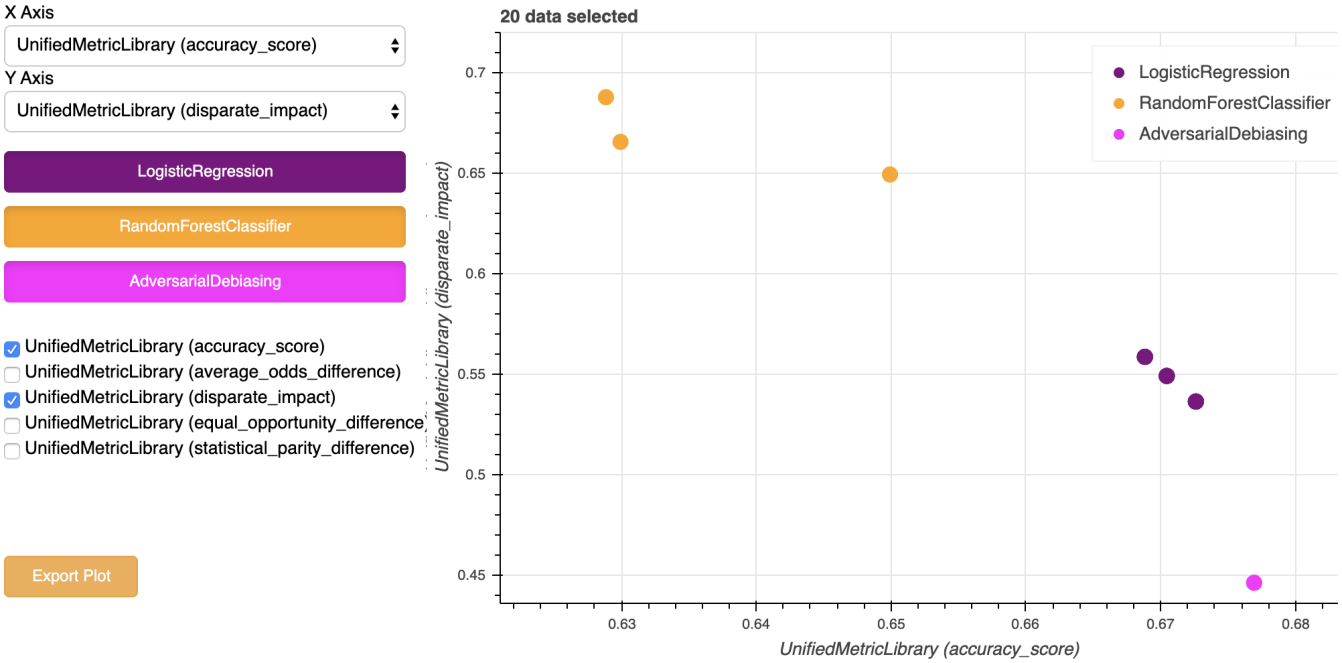

Figure 3: Fairkit learn's output visualization for the search parameters in Figure 1.

Fairkit-learn supports all of scikit-learn's and AIF360's algorithms and metrics, including all of AIF360's bias mitigating algorithms. Fairkit-learn is currently capable of working with over 70 definition of fairness, including:

- **Disparate impact**, which measures if a model treats similarly the same fraction of individuals of each group [10, 14, 37].
- **Demographic parity**, also called **statistical parity** and **group fairness**, which measures if a model's predictions are statistically independent of the attribute with respect to which the model is fair [8, 11].
- **Equal opportunity difference**, which requires that false negative rates among groups are equal [10, 15].
- **Causal fairness**, also called **counterfactual fairness**, requires classifiers to predict the same outcome for two individuals that s differ only in protected attributes, and are otherwise identical [12, 23].

Fairkit-learn also provides extension points for including additional metrics and algorithms. Fairkit-learn builds on the contributions of scikit-learn and AI Fairness 360 by providing the following unique features, which we discuss in more detail next:

- An automated model search capable of evaluating thousands of machine learning models with respect to two quality and/or fairness metrics simultaneously.
- An interactive visualization that allows users to explore and compare a small, Pareto-optimal set of models for each set of metrics selected.

Model search. Unlike existing tools, fairkit-learn allows users to find models that best balance fairness with other quality concerns across as many models as the user can computationally support (the more models you want to assess, the more memory and power needed). Figure 1 shows an example set of inputs used for fairkit-learn's `ModelSearch`. The required inputs for fairkit-learn's model search are:

- **Models.** Fairkit-learn requires the specification of at least one model (*models* in Figure 1) to perform the grid search, but users can specify as many models as their computational resources will allow.
- **Metrics.** Along with models, the user must specify the metrics fairkit-learn should use to evaluate each model configuration (*metrics* in Figure 1). All of fairkit-learn's current metrics are stored in the `UnifiedMetricLibrary`.
- **Hyperparameters.** This is an optional input, where the user can specify value ranges for each hyperparameter of each model (*hyperparameters* in Figure 1). If no hyperparamter ranges are specified, fairkit-learn will use default model configurations in the grid search.
- **Thresholds.** Users need to specify the probabilistic threshold required for positive binary classification (*thresholds* in Figure 1) . For example, a threshold of 0.8 will consider any prediction with ≥ 0.7 probability to be favorable.
- **Pre- and post-processing algorithms.** Lastly, users can add any pre- and post-processing algorithms to include in the grid search (*preprocessors* and *postprocessors* in Figure 1).

Once the search is done, results are written to a .csv file that is used to render a visualization of the results from the grid search.

Search visualization. One of the ways that users can process the grid search results are presented through the interactive visualization provided by fairkit-learn, as shown in Figure 3. The plot that is shown was rendered using the results from the search in Figure 1. The fairkit-learn visualization allows for viewing of Pareto-optimal models for any two metrics by selecting those metrics from the checklist in the bottom left corner of Figure 3 and choosing those metrics as the X and Y axes in the dropboxes in the upper left corner. Users can also access all of the search results, including non-optimal models, by selecting all the metrics in the checklist. To toggle between models and metrics being displayed in the visualization, users can select the X and Y axes along with selecting (or de-selecting) models to include using the different color model buttons (e.g., the magenta `AdversarialDebiasing` button). Users can export the results of their search as a plot that comes with a JSON file that describes the plot.

3 USING FAIRKIT-LEARN TO FIND OPTIMAL MODELS

Fairkit-learn is an open source Python toolkit that supports interactive evaluation and comparison of machine learning models for fairness and other quality metrics simultaneously. It can evaluate thousands of models produced by multiple machine learning algorithms, hyperparameters, and data permutations, and compute then visualize a small Pareto-optimal set of models that provide an optimal balance between fairness and quality. Data scientists can then iterate, improving their models and evaluating them using fairkit-learn. Instructions for installing fairkit-learn, along with a tutorial implemented in a Jupyter notebook, can be found here: https://go.gmu.edu/fairkit-learn

To better understand how a data scientist could use fairkit-learn to train, evaluate, and compare machine learning models, let us look back at the search written in Figure 1. Here, the user wants to compare the resulting models from three algorithms: `LogisticRegression`, `RandomForestClassifier`, and `AdversarialDebiasing`. In this example, she is using the COMPAS recidivism dataset, which contains recidivism data for Browards County between the years 2013 and 2014 [28]. Let us imagine that the user wants to explore models that best balance accuracy and fairness; one could choose any metric for each of these concerns, however, she cares about the *accuracy_score* and the *disparate_impact*.

While the user has only entered three learning algorithms, fairkit-learn will train approximately 80 different models by using the hyperparameter value ranges specified to vary the hyperparameter values in the grid-search. The grid-search will produce a substantially smaller subset of models (in this case, the output includes only 7 models) that make up the Pareto-optimal set. This smaller set of models is then presented to the user in a visualization that plots the models with respect to the metrics chosen by the user; in this case, we have disparate impact (a fairness metric) on the x-axis and accuracy (a quality metric) on the y-axis.

The final set of optimal models shown in the visualization are models for which improving fairness would decrease accuracy and vice versa (the Pareto-optimal set). For the search shown in Figure 1, the interactive visualization in Figure 3 makes it easy to come realize that (1) given this dataset (COMPAS), model fairness and accuracy are often in opposition – in other domains they may be complementary, (2) we can achieve a large increase in fairness (69% compared to 45%) if we sacrifice some accuracy (63% compared to 68%), (3) while adversarial debiasing (magenta) is a fairness-aware algorithm, it produces less fair but slights more accurate models in comparison to random forest classifier models (orange) that produce more fair models with a small decrease in accuracy, and (4) with respect to both accuracy and fairness, logistic regression models (purple) tend to be more balanced than the other two algorithms with accuracy that is comparable to adversarial debiasing but increased fairness.

Users can dig deeper into the search results by *hovering over the data points* in the plot to get more information on the Pareto-optimal models found (e.g., the hyperparameter values for that configuration), *opening the .csv* file used to render the visualization, which includes all models included in the search, or *exporting the plot* and using the accompanying JSON file to examine optimal models.

User Evaluation. To evaluate the potential for fairkit-learn to be useful in practice, we conducted a user study with 54 graduate and undergraduate students with varying experience training machine learning models [20]. We asked participants to complete a series of tasks involving training and evaluating machine learning models for fairness and accuracy. Along with using fairkit-learn, we asked participants to complete the same tasks using scikit-learn, the state-of-the-art in training machine learning models, and AIF360, one of them more recognized model fairness evaluation toolkits. We found that participants selected fairer, more accurate models when using fairkit-learn. When trying to balance fairness and accuracy, fairkit-learn was able to find models that are high performing and generally more fair than the models found by AIF360 and scikit-learn.

4 RELATED WORK

Typically, machine learning model performance is evaluated using accuracy metrics. scikit-learn [26], one of the most common tools used for training and evaluating machine learning models, provides engineers with a variety of machine learning algorithms and various metrics for evaluating models for performance. While scikit-learn is useful for training and evaluating models based on their performance, there is no built in functionality for measuring model fairness or mitigating bias. But fairness of machine learning models plays an important role in software that uses such models [5].

There do exist tools designed to help engineers reason about fairness in their machine learning models [1, 3, 36]. FairML supports the detection of unintended discrimination in predictive models by automatically determining the relative significance of model inputs to outcomes [1]. Another solution, Fairway, combines pre-processing and in-processing algorithms to remove bias from training data and models [9].

Fairkit-learn supports evaluating models with respect to two metrics simultaneously via an interactive visualization of Pareto-optimal models. Most related to our contribution is Microsoft's Fairlearn, a Python toolkit that uses interactive visualizations to support evaluating model fairness and fairness-performance trade-offs [4]. While both fairkit-learn and Fairlearn provide similar functionality to accomplish similar goals, fairkit-learn has a couple of unique features. First, fairkit-learn was built using state-of-the-art machine learning and fairness libraries increasing ease of integration into existing workflows. Also, currently Fairlearn is only capable of measuring group discrimination while fairkit-learn supports any definition of fairness in AI Fairness 360 (or that the user would like to add or implement). Lastly, fairkit-learn only returns Pareto-optimal models.

Google developed the What-If Tool to support the analysis and understanding of machine learning models without having to write code [36]. When given a TensorFlow model and a dataset, the What-If Tool visualizes the dataset, allows for editing of individuals in the dataset, shows the effects of dataset modification, performs counterfactual analysis, and evaluates models based on performance and fairness. Fairea, a model behavior mutation approach, is another intervention that supports measuring and evaluating fairness-accuracy trade-offs when using machine learning bias mititgation methods [18].

AI Fairness 360 (AIF360), a Python tool suite for evaluating model fairness and performance [3], includes fairness metrics, metric explanations, and bias mitigation algorithms for datasets and models. AIF360 is designed to be extensible and accessible to data scientists and practitioners. Also similar is FairVis, a visual analytics system that supports exploring fairness and performance with respect to certain subgroups in a dataset [7]. Similar to fairkit-learn, FairVis uses visualizations to support this exploration.

Some machine learning methods, known as Seldonian algorithms, provide high-confidence guarantees that learned models enforce user-specified fairness properties, even when applied to unseen data [24, 34]. These guarantees can even extend to settings when the distribution of the training data is different from that of the data to which the model is applied [13].

There also exist tools designed to support engineers ability to test their software for fairness [2, 12, 31, 35, 38, 39]. Themis, a software fairness testing tool, was the first of its kind [2, 12]. Themis automatically generates tests that help engineers detect and measure causal and group discrimination. Fairness testing can be made more efficient in finding inputs that exhibit bias [35, 38, 39], and can be driven by a grammar [31].

While there exists tools that can help engineers evaluate models for fairness and performance and measure software bias, fairkit-learn works with existing tools to help engineers find Pareto-optimal models that balance fairness and performance and provides an interactive visualization that makes it quicker and easier to explore the effects of different model configurations.

5 CONTRIBUTIONS

We presented fairkit-learn, a novel open-source toolkit designed to support the evaluation and comparison of machine learning models across multiple dimensions, such as fairness and performance. We described how to use fairkit-learn to train, evaluate, and compare machine learning models using its interactive visualization interface. We outline the potential for fairkit-learn to be beneficial in practice based on results from a controlled user study, demonstrating that fairkit-learn is an effective tool for helping data scientists understand the fairness-quality landscape.

ACKNOWLEDGMENTS

Jesse Bartola, Rico Angell, Sam Witty, Stephen J. Giguere, and Katherine A. Keith helped build and evaluate fairkit-learn. This work is supported by the National Science Foundation under grant no CCF-1763423, and by Google, Meta Platforms, and Kosa.ai.

REFERENCES

[1] Julius A. Adebayo. 2016. *FairML: ToolBox for diagnosing bias in predictive modeling*. Ph. D. Dissertation. Massachusetts Institute of Technology.

[2] Rico Angell, Brittany Johnson, Yuriy Brun, and Alexandra Meliou. 2018. Themis: Automatically testing software for discrimination. In *Proceedings of the 2018 26th ACM Joint meeting on european software engineering conference and symposium on the foundations of software engineering*. 871–875.

[3] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. 2018. AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias. *CoRR* 1810.01943 (2018). https://arxiv.org/abs/1810.01943

[4] Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. 2020. *Fairlearn: A toolkit for assessing and improving fairness in AI*. Technical Report MSR-TR-2020-32. Microsoft.

[5] Yuriy Brun and Alexandra Meliou. 2018. Software Fairness. In *Proceedings of the New Ideas and Emerging Results Track at the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)* (6–9). Lake Buena Vista, FL, USA, 754–759. https://doi.org/10.1145/3236024.3264838

[6] Joy Buolamwini and Timnit Gebru. 2018. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, Vol. 81. PMLR, New York, NY, USA, 77–91. http://proceedings.mlr.press/v81/buolamwini18a.html

[7] Ángel Alexander Cabrera, Will Epperson, Fred Hohman, Minsuk Kahng, Jamie Morgenstern, and Duen Horng Chau. 2019. FairVis: Visual analytics for discovering intersectional bias in machine learning. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*. 46–56.

[8] Toon Calders and Sicco Verwer. 2010. Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery* 21, 2 (2010), 277–292. https://doi.org/10.1007/s10618-010-0190-x

[9] Joymallya Chakraborty, Suvodeep Majumder, Zhe Yu, and Tim Menzies. 2020. Fairway: A Way to Build Fair ML Software. In *ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*.

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

[11] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Innovations in Theoretical Computer Science Conference (ITCS)*. Cambridge, MA, USA, 214–226.

[12] Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou. 2017. Fairness testing: Testing software for discrimination. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. 498–510.

[13] Stephen Giguere, Blossom Metevier, Yuriy Brun, Philip S. Thomas, Scott Niekum, and Bruno Castro da Silva. 2022. Fairness Guarantees under Demographic Shift. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)* (25–29).

[14] Griggs v. Duke Power Co. 1971. 401 U.S. 424. https://supreme.justia.com/cases/federal/us/401/424/.

[15] Moritz Hardt, Eric Price, and Nathan Srebro. 2016. Equality of Opportunity in Supervised Learning. In *Annual Conference on Neural Information Processing Systems (NIPS)*. Barcelona, Spain.

[16] Devindra Haweawar. 2012. Staples, Home Depot, and other online stores change prices based on your location. *VentureBeat* December 24 (2012). https://venturebeat.com/2012/12/24/staples-online-stores-price-changes.

[17] Kashmir Hill. 2020. Wrongfully Accused by an Algorithm. *The New York Times* August 3 (2020). nytimes.com/2020/06/24/technology/facial-recognition-arrest.html.

[18] Max Hort, Jie M. Zhang, Federica Sarro, and Mark Harman. 2021. Fairea: A Model Behaviour Mutation Approach to Benchmarking Bias Mitigation Methods. In *European Software Engineering Conference and ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE)*. Athens, Greece, 994–1006. https://doi.org/10.1145/3468264.3468565

[19] IBM. 2019. AI Fairness 360 Open Source Toolkit. https://aif360.mybluemix.net.

[20] Brittany Johnson, Jesse Bartola, Rico Angell, Katherine Keith, Sam Witty, Stephen J Giguere, and Yuriy Brun. 2020. Fairkit, Fairkit, on the Wall, Who's the Fairest of Them All? Supporting Data Scientists in Training Fair Models. *arXiv preprint arXiv:2012.09951* (2020).

[21] Brendan F. Klare, Mark J. Burge, Joshua C. Klontz, Richard W. Vorder Bruegge, and Anil K. Jain. 2012. Face Recognition Performance: Role of Demographic Information. *IEEE Transactions on Information Forensics and Security (TIFS)* 7, 6 (December 2012), 1789–1801. https://doi.org/10.1109/TIFS.2012.2214212

[22] Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R. Rickford, Dan Jurafsky, and Sharad Goel. 2020. Racial disparities in automated speech recognition. *Proceedings of the National Academy of Sciences* 117, 14 (2020), 7684–7689. https://doi.org/10.1073/pnas.1915768117

[23] Matt J. Kusner, Joshua R. Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual Fairness. In *Annual Conference on Neural Information Processing Systems (NIPS)*. Long Beach, CA, USA.

[24] Blossom Metevier, Stephen Giguere, Sarah Brockman, Ari Kobren, Yuriy Brun, Emma Brunskill, and Philip Thomas. 2019. Offline Contextual Bandits with High Probability Fairness Guarantees. In *33rd Annual Conference on Neural Information Processing Systems (NeurIPS), Advances in Neural Information Processing Systems 32* (9–14). Vancouver, BC, Canada, 14893–14904. http://papers.neurips.cc/paper/9630-offline-contextual-bandits-with-high-probability-fairness-guarantees

[25] Jakub Mikians, László Gyarmati, Vijay Erramilli, and Nikolaos Laoutaris. 2012. Detecting Price and Search Discrimination on the Internet. In *ACM Workshop on Hot Topics in Networks (HotNets)*. Redmond, Washington, 79–84. https://doi.org/10.1145/2390231.2390245

[26] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[27] Tony Peng. 2019. Humans Don't Realize How Biased They Are Until AI Reproduces the Same Bias, Says UNESCO AI Chair. https://tinyurl.com/y5jxadg6/.

[28] ProPublica. 2019. COMPAS Recidivism Risk Score Data and Analysis. https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis/.

[29] Manish Raghavan, Solon Barocas, Jon Kleinberg, and Karen Levy. 2019. Mitigating Bias in Algorithmic Hiring: Evaluating Claims and Practices. *CoRR* 1906.09208 (2019). https://arxiv.org/abs/1906.09208

[30] scikit-learn 2019. scikit-learn: Machine Learning in Python. https://scikit-learn.org/stable/.

[31] Ezekiel Soremekun, Sakshi Udeshi, and Sudipta Chattopadhyay. 2022. AS-TRAEA: Grammar-based Fairness Testing. *IEEE Transactions on Software Engineering (TSE)* (Jan. 2022), 24. https://doi.org/10.1109/TSE.2022.3141758

[32] Eliza Strickland. 2016. Doc bot preps for the O.R. *IEEE Spectrum* 53, 6 (June 2016), 32–60. https://doi.org/10.1109/MSPEC.2016.7473150

[33] Rachael Tatman. 2017. Gender and Dialect Bias in YouTube's Automatic Captions. In *Workshop on Ethics in Natural Language Processing*. Valencia, Spain. https://doi.org/10.18653/v1/W17-1606

[34] Philip S. Thomas, Bruno Castro da Silva, Andrew G. Barto, Stephen Giguere, Yuriy Brun, and Emma Brunskill. 2019. Preventing Undesirable Behavior of Intelligent Machines. *Science* 366, 6468 (22 November 2019), 999–1004. https://doi.org/10.1126/science.aag3311

[35] Sakshi Udeshi, Pryanshu Arora, and Sudipta Chattopadhyay. 2018. *Automated Directed Fairness Testing*. Montpellier, France, 98–108. https://doi.org/10.1145/3238147.3238165

[36] James Wexler. 2018. The What-If Tool: Code-Free Probing of Machine Learning Models. https://ai.googleblog.com/2018/09/the-what-if-tool-code-free-probing-of.html.

[37] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P. Gummadi. 2017. Fairness beyond disparate treatment & Disparate impact: Learning classification without disparate mistreatment. In *Fairness, Accountability, and Transparency in Machine Learning (FAT ML)*. Perth, Australia.

[38] Lingfeng Zhang, Yueling Zhang, and Min Zhang. 2021. Efficient White-Box Fairness Testing through Gradient Search. In *International Symposium on Software Testing and Analysis (ISSTA)*. Association for Computing Machinery, Denmark, 103–114. https://doi.org/10.1145/3460319.3464820

[39] Peixin Zhang, Jingyi Wang, Jun Sun, Guoliang Dong, Xinyu Wang, Xingen Wang, Jin Song Dong, and Ting Dai. 2020. White-Box Fairness Testing through Adversarial Sampling. In *ACM/IEEE International Conference on Software Engineering (ICSE)*. Seoul, South Korea, 949–960. https://doi.org/10.1145/3377811.3380331