Engineering Software Fairness: What Is Still Missing?

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

DISIM
Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

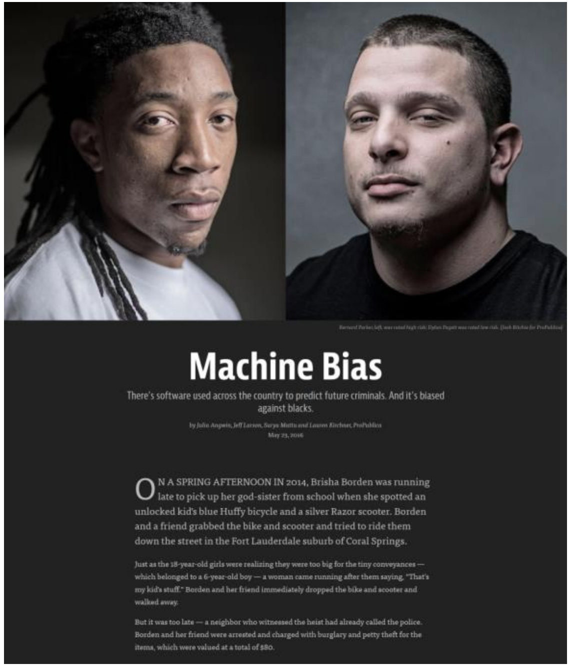
Giordano d'Aloisio

Università degli Studi dell'Aquila / Italy

SSE Seminar, University College London
24 April 2024

COMPAS

- COMPAS is an ML algorithm used by some courts in the US to predict recidivism of condemned people
- A study showed that, given two people with the same features but different race, the system was giving higher probability of recidivism to non-white people



Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica
May 23, 2016

ON A SPRING AFTERNOON IN 2014, Brisha Borden was running late to pick up her god-sister from school when she spotted an unlocked kid's blue Huffy bicycle and a silver Razor scooter. Borden and a friend grabbed the bike and scooter and tried to ride them down the street in the Fort Lauderdale suburb of Coral Springs.

Just as the 18-year-old girls were realizing they were too big for the tiny conveyances — which belonged to a 6-year-old boy — a woman came running after them saying, "That's my kid's stuff." Borden and her friend immediately dropped the bike and scooter and walked away.

But it was too late — a neighbor who witnessed the heist had already called the police. Borden and her friend were arrested and charged with burglary and petty theft for the items, which were valued at a total of $80.

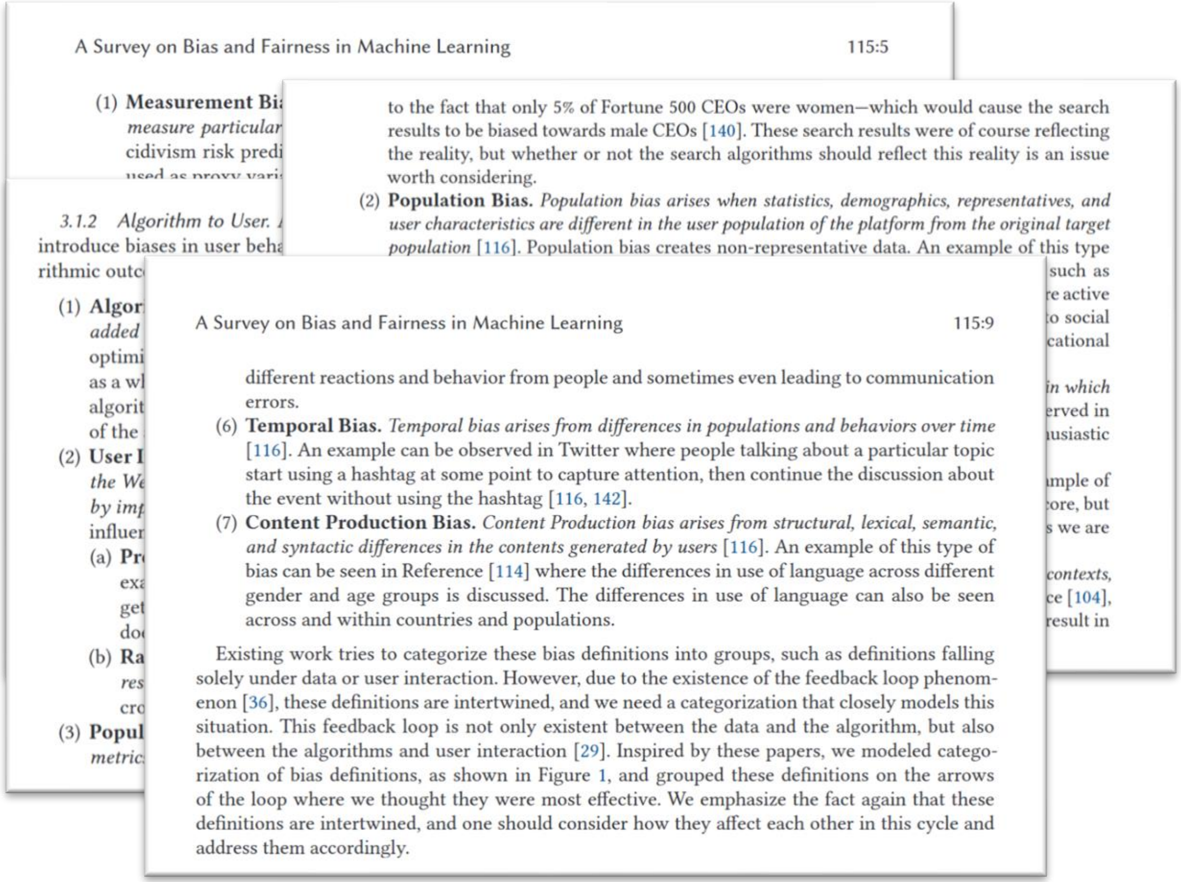
The system was biased against non-white people

Bias and Fairness

- *BIAS:* systematic favouritism or discrimination in models' predictions towards individuals based on some sensitive features (like *gender*, *race*, and others)
- *FAIRNESS:* absence of favouritism or discrimination in models' predictions

N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, 'A Survey on Bias and Fairness in Machine Learning', ACM Comput. Surv., vol. 54, no. 6, pp. 1–35, Jul. 2021, doi: 10.1145/3457607.

Is the Concept of Bias that Simple?



(1) **Measurement Bi**
measure particular
cidivism risk predi
used as proxy vari

3.1.2 Algorithm to User.
introduce biases in user beha
rithmic outc

(1) **Algor**
added
optimi
as a wl
algorit
of the
(2) **User I**
the We
by imp
influer
(a) **Pr**
exa
get
do
(b) **Ra**
res
cro
(3) **Popul**
metric

to the fact that only 5% of Fortune 500 CEOs were women—which would cause the search results to be biased towards male CEOs [140]. These search results were of course reflecting the reality, but whether or not the search algorithms should reflect this reality is an issue worth considering.

(2) **Population Bias.** *Population bias arises when statistics, demographics, representatives, and user characteristics are different in the user population of the platform from the original target population* [116]. Population bias creates non-representative data. An example of this type
such as
re active
o social
cational

in which
erved in
usiastic

mple of
ore, but
s we are

contexts,
ce [104],
result in

different reactions and behavior from people and sometimes even leading to communication errors.

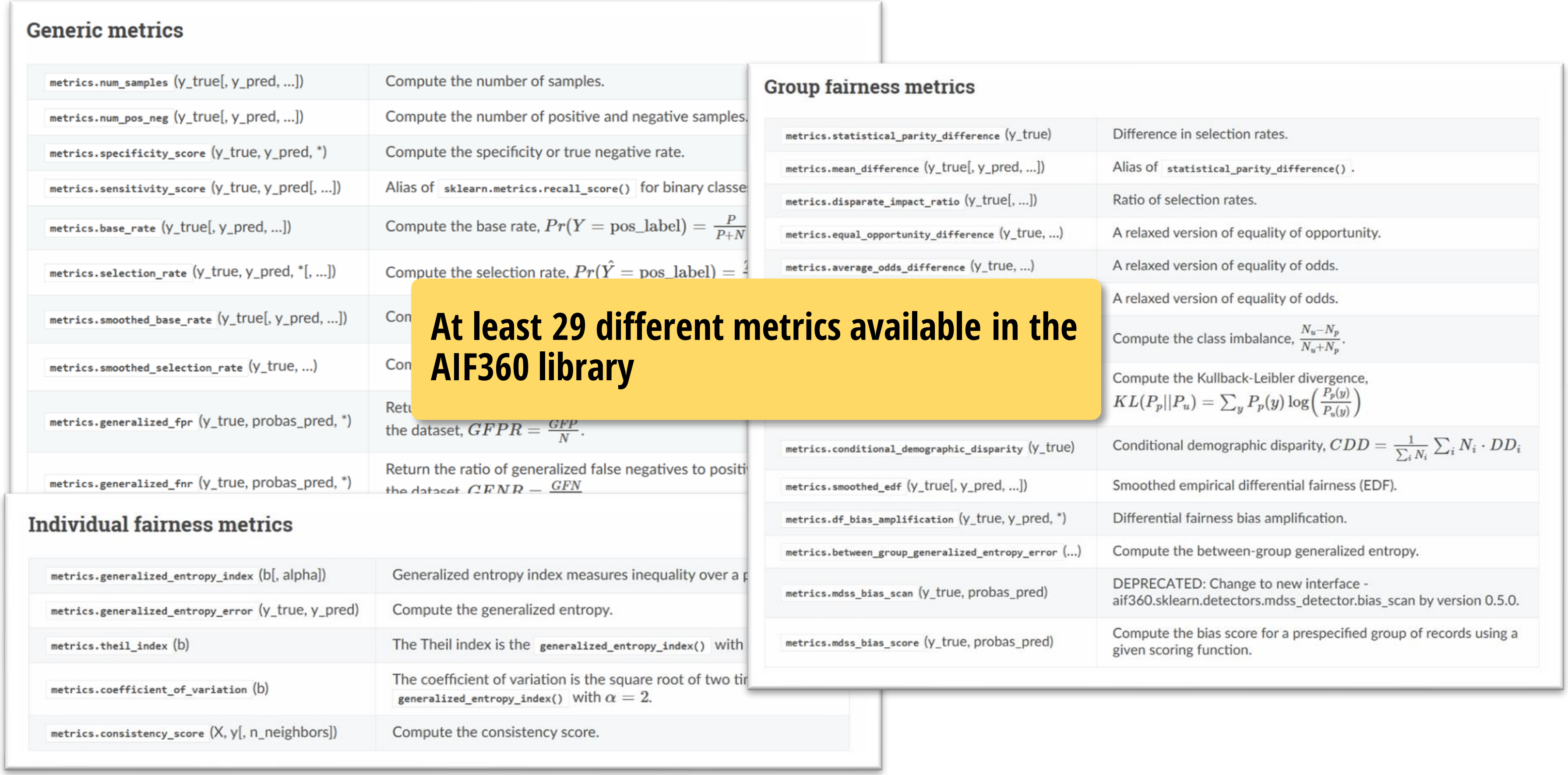
(6) **Temporal Bias.** *Temporal bias arises from differences in populations and behaviors over time* [116]. An example can be observed in Twitter where people talking about a particular topic start using a hashtag at some point to capture attention, then continue the discussion about the event without using the hashtag [116, 142].

(7) **Content Production Bias.** *Content Production bias arises from structural, lexical, semantic, and syntactic differences in the contents generated by users* [116]. An example of this type of bias can be seen in Reference [114] where the differences in use of language across different gender and age groups is discussed. The differences in use of language can also be seen across and within countries and populations.

Existing work tries to categorize these bias definitions into groups, such as definitions falling solely under data or user interaction. However, due to the existence of the feedback loop phenomenon [36], these definitions are intertwined, and we need a categorization that closely models this situation. This feedback loop is not only existent between the data and the algorithm, but also between the algorithms and user interaction [29]. Inspired by these papers, we modeled categorization of bias definitions, as shown in Figure 1, and grouped these definitions on the arrows of the loop where we thought they were most effective. We emphasize the fact again that these definitions are intertwined, and one should consider how they affect each other in this cycle and address them accordingly.

At least 23 different definitions of bias in the literature

From Many Definitions Come Many Metrics



Generic metrics

`metrics.num_samples` (y_true[, y_pred, ...])	Compute the number of samples.
`metrics.num_pos_neg` (y_true[, y_pred, ...])	Compute the number of positive and negative samples.
`metrics.specificity_score` (y_true, y_pred, *)	Compute the specificity or true negative rate.
`metrics.sensitivity_score` (y_true, y_pred[, ...])	Alias of `sklearn.metrics.recall_score()` for binary classe
`metrics.base_rate` (y_true[, y_pred, ...])	Compute the base rate, $Pr(Y = \text{pos\_label}) = \frac{P}{P+N}$
`metrics.selection_rate` (y_true, y_pred, *[, ...])	Compute the selection rate, $Pr(\hat{Y} = \text{pos\_label}) =$
`metrics.smoothed_base_rate` (y_true[, y_pred, ...])	Con
`metrics.smoothed_selection_rate` (y_true, ...)	Con
`metrics.generalized_fpr` (y_true, probas_pred, *)	Retu the dataset, $GFPR = \frac{GFP}{N}$.
`metrics.generalized_fnr` (y_true, probas_pred, *)	Return the ratio of generalized false negatives to positi the dataset, $GFNR = \frac{GFN}{}$

At least 29 different metrics available in the AIF360 library

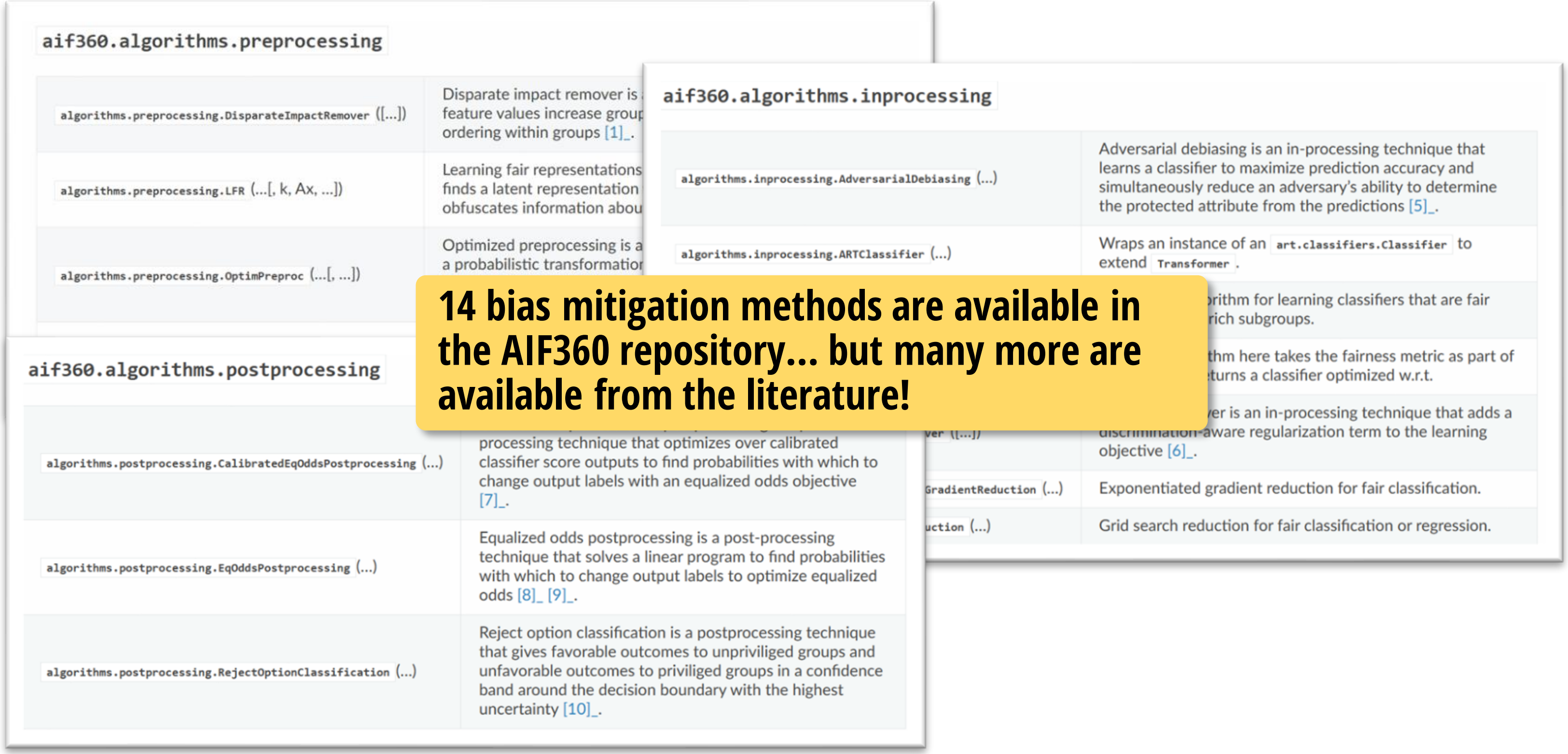
Group fairness metrics

`metrics.statistical_parity_difference` (y_true)	Difference in selection rates.
`metrics.mean_difference` (y_true[, y_pred, ...])	Alias of `statistical_parity_difference()` .
`metrics.disparate_impact_ratio` (y_true[, ...])	Ratio of selection rates.
`metrics.equal_opportunity_difference` (y_true, ...)	A relaxed version of equality of opportunity.
`metrics.average_odds_difference` (y_true, ...)	A relaxed version of equality of odds.
	A relaxed version of equality of odds.
	Compute the class imbalance, $\frac{N_u - N_p}{N_u + N_p}$.
	Compute the Kullback-Leibler divergence, $KL(P_p \| P_u) = \sum_y P_p(y) \log\left(\frac{P_p(y)}{P_u(y)}\right)$
`metrics.conditional_demographic_disparity` (y_true)	Conditional demographic disparity, $CDD = \frac{1}{\sum_i N_i} \sum_i N_i \cdot DD_i$
`metrics.smoothed_edf` (y_true[, y_pred, ...])	Smoothed empirical differential fairness (EDF).
`metrics.df_bias_amplification` (y_true, y_pred, *)	Differential fairness bias amplification.
`metrics.between_group_generalized_entropy_error` (...)	Compute the between-group generalized entropy.
`metrics.mdss_bias_scan` (y_true, probas_pred)	DEPRECATED: Change to new interface - aif360.sklearn.detectors.mdss_detector.bias_scan by version 0.5.0.
`metrics.mdss_bias_score` (y_true, probas_pred)	Compute the bias score for a prespecified group of records using a given scoring function.

Individual fairness metrics

`metrics.generalized_entropy_index` (b[, alpha])	Generalized entropy index measures inequality over a p
`metrics.generalized_entropy_error` (y_true, y_pred)	Compute the generalized entropy.
`metrics.theil_index` (b)	The Theil index is the `generalized_entropy_index()` with
`metrics.coefficient_of_variation` (b)	The coefficient of variation is the square root of two tir `generalized_entropy_index()` with $\alpha = 2$.
`metrics.consistency_score` (X, y[, n_neighbors])	Compute the consistency score.

Mitigating Bias



`aif360.algorithms.preprocessing`

`algorithms.preprocessing.DisparateImpactRemover` ([...])	Disparate impact remover is feature values increase group ordering within groups [1]_.
`algorithms.preprocessing.LFR` (...[, k, Ax, ...])	Learning fair representations finds a latent representation obfuscates information abou
`algorithms.preprocessing.OptimPreproc` (...[, ...])	Optimized preprocessing is a a probabilistic transformation

`aif360.algorithms.inprocessing`

`algorithms.inprocessing.AdversarialDebiasing` (...)	Adversarial debiasing is an in-processing technique that learns a classifier to maximize prediction accuracy and simultaneously reduce an adversary's ability to determine the protected attribute from the predictions [5]_.
`algorithms.inprocessing.ARTClassifier` (...)	Wraps an instance of an `art.classifiers.Classifier` to extend `Transformer` .
	rithm for learning classifiers that are fair rich subgroups.
	thm here takes the fairness metric as part of turns a classifier optimized w.r.t.
`ver` ([...])	er is an in-processing technique that adds a discrimination-aware regularization term to the learning objective [6]_.
`GradientReduction` (...)	Exponentiated gradient reduction for fair classification.
`uction` (...)	Grid search reduction for fair classification or regression.

`aif360.algorithms.postprocessing`

`algorithms.postprocessing.CalibratedEqOddsPostprocessing` (...)	processing technique that optimizes over calibrated classifier score outputs to find probabilities with which to change output labels with an equalized odds objective [7]_.
`algorithms.postprocessing.EqOddsPostprocessing` (...)	Equalized odds postprocessing is a post-processing technique that solves a linear program to find probabilities with which to change output labels to optimize equalized odds [8]_ [9]_.
`algorithms.postprocessing.RejectOptionClassification` (...)	Reject option classification is a postprocessing technique that gives favorable outcomes to unprivileged groups and unfavorable outcomes to priviliged groups in a confidence band around the decision boundary with the highest uncertainty [10]_.

14 bias mitigation methods are available in the AIF360 repository… but many more are available from the literature!

What is Missing?

Challenge 1

Most bias mitigation methods address binary classification. What about multi-class classification?

Challenge 2

How can we guide a user non-expert about fairness in making fairness evaluations?

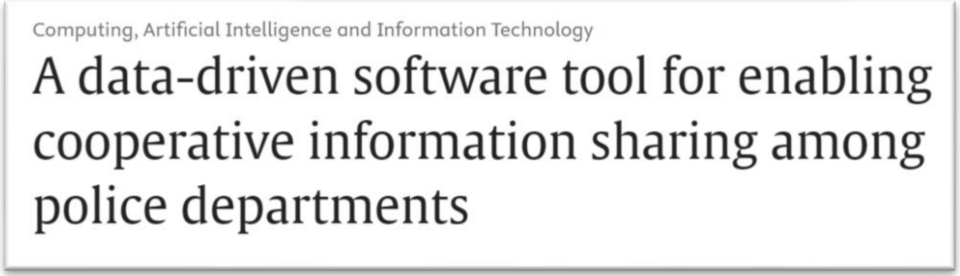
Challenge 3

Most fairness assessment approaches are domain and definition specific. How can we address non-traditional use cases?

Engineering Software Fairness

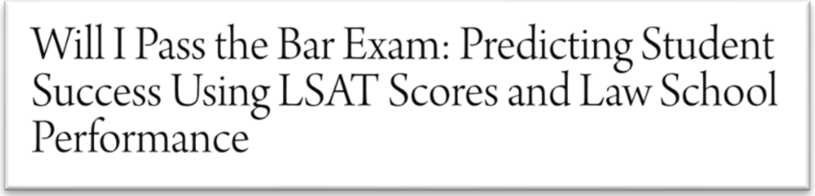
Challenge 1: Bias in Multi-Class Classification

- Most of the bias mitigation approaches focus on binary classification
- However, many multi-class classification approaches have been proposed in sensitive domains

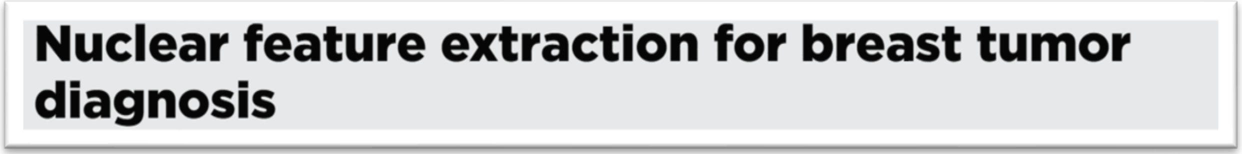


Computing, Artificial Intelligence and Information Technology

A data-driven software tool for enabling cooperative information sharing among police departments



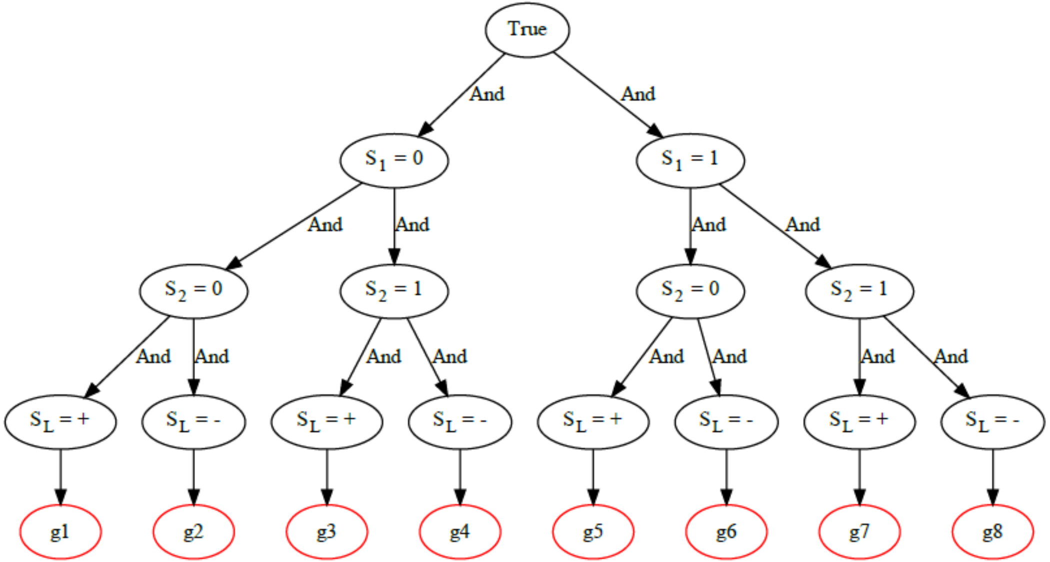
Will I Pass the Bar Exam: Predicting Student Success Using LSAT Scores and Law School Performance



Nuclear feature extraction for breast tumor diagnosis

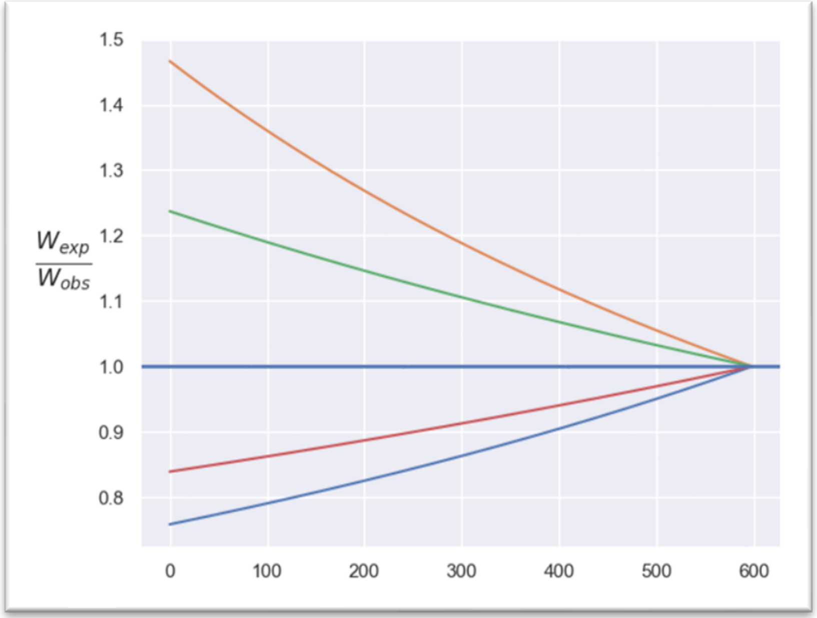
Addressing Challenge 1: DEMV [1-2]

- DEMV is a pre-processing approach to improve fairness in binary and multi-class classification tasks
- First identifies all the *sensitive groups* given by the combination of sensitive variables and label's values

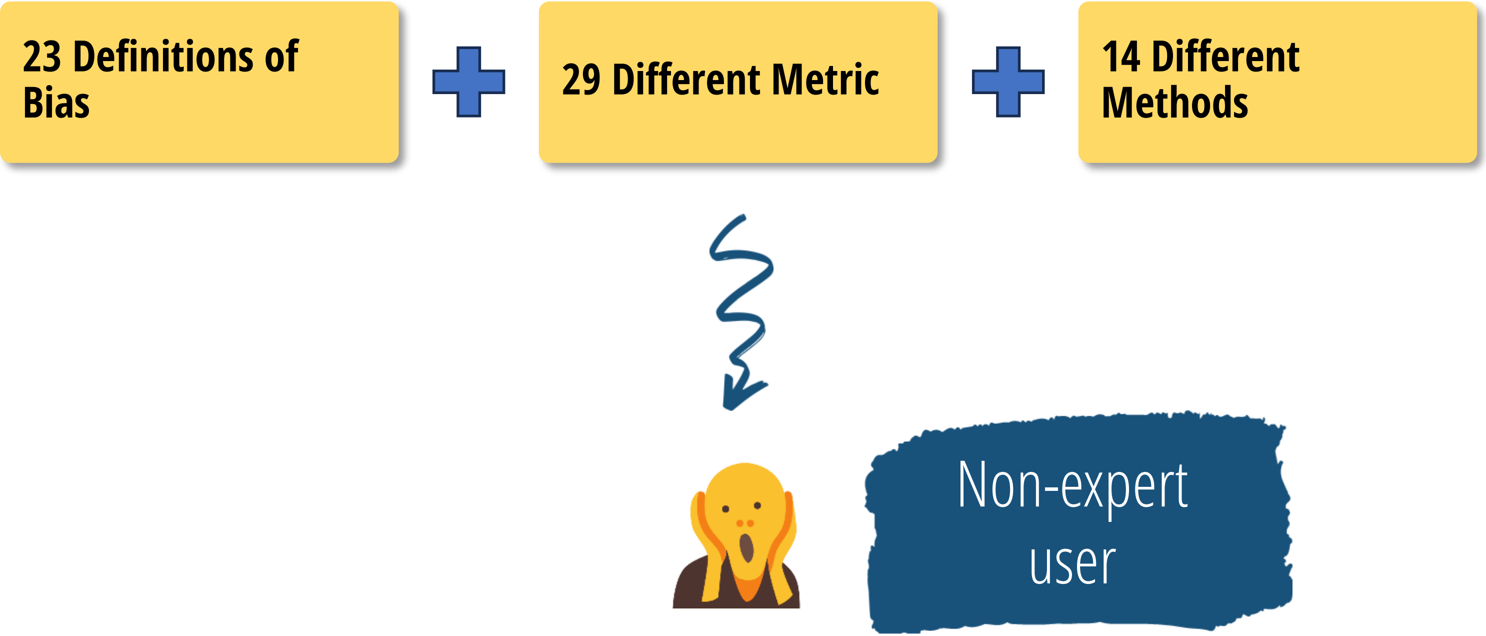


Debiaser for Multiple Variables (DEMV) [1-2]

- Next, rebalance the sensitive groups until the observed size is equal to the expected one
- Overcomes all the other state-of-the-art multi-class bias mitigation algorithms in the literature
- Algorithm available online and on pip:



Challenge 2: Democratising Software Fairness



23 Definitions of Bias

29 Different Metric

14 Different Methods

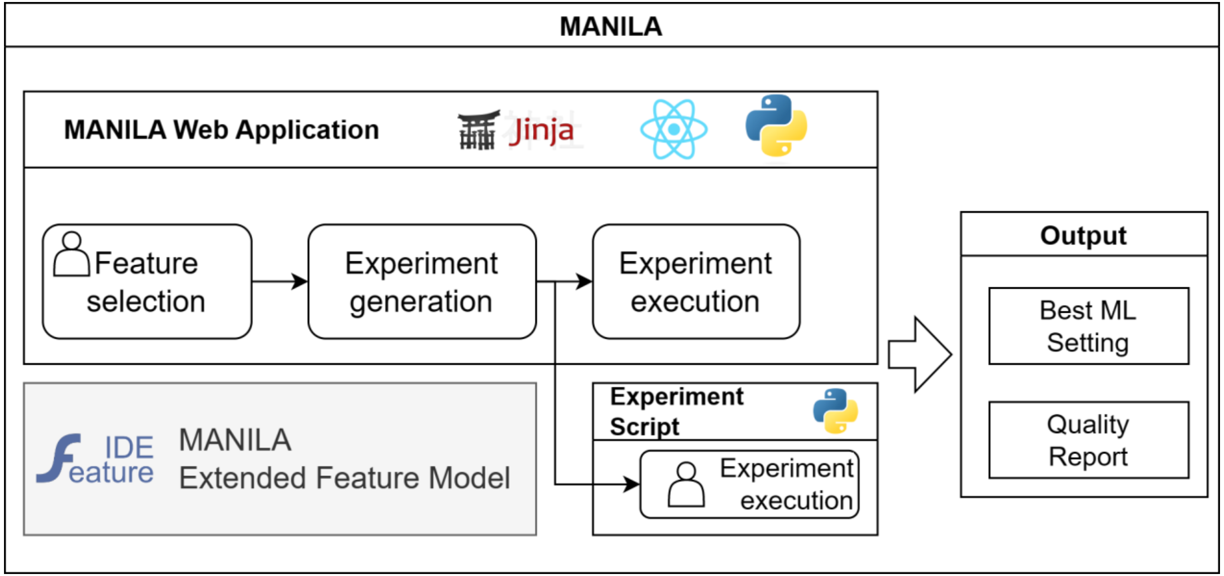
Non-expert user

Addressing Challenge 2: MANILA [3]

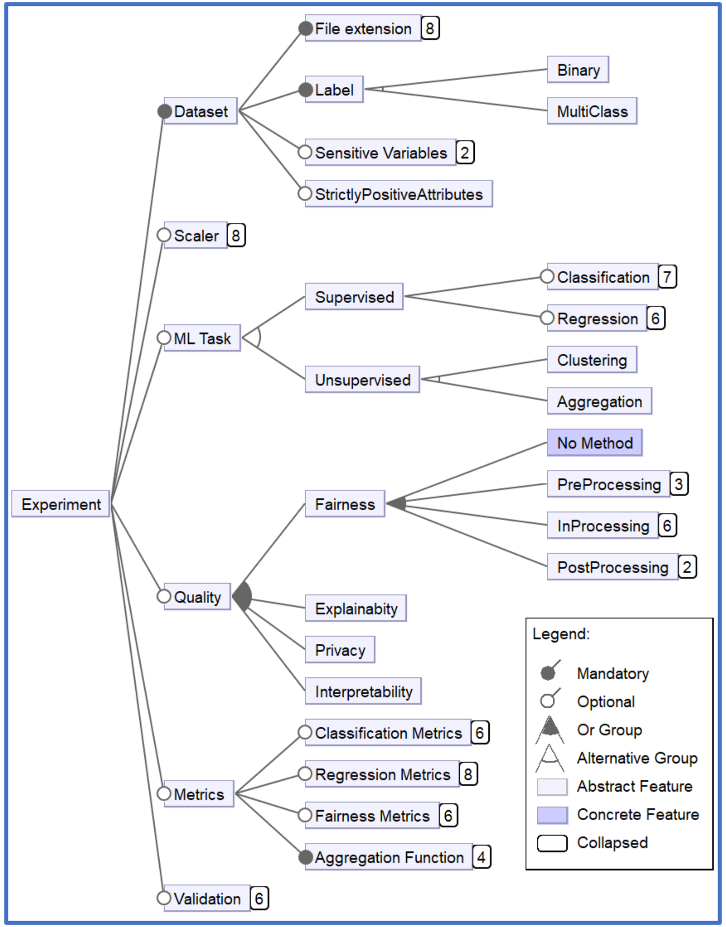
- We propose MANILA, a web-based application to design, implement and execute fairness evaluations
- Think of a fairness evaluation workflow as a Software Product Line (SPA)
- The variation points are ML methods, Fairness methods, Metrics, ...
- The constraints in SPA guide users in creating evaluations that are always executable
- Available in the SoBigData RI:

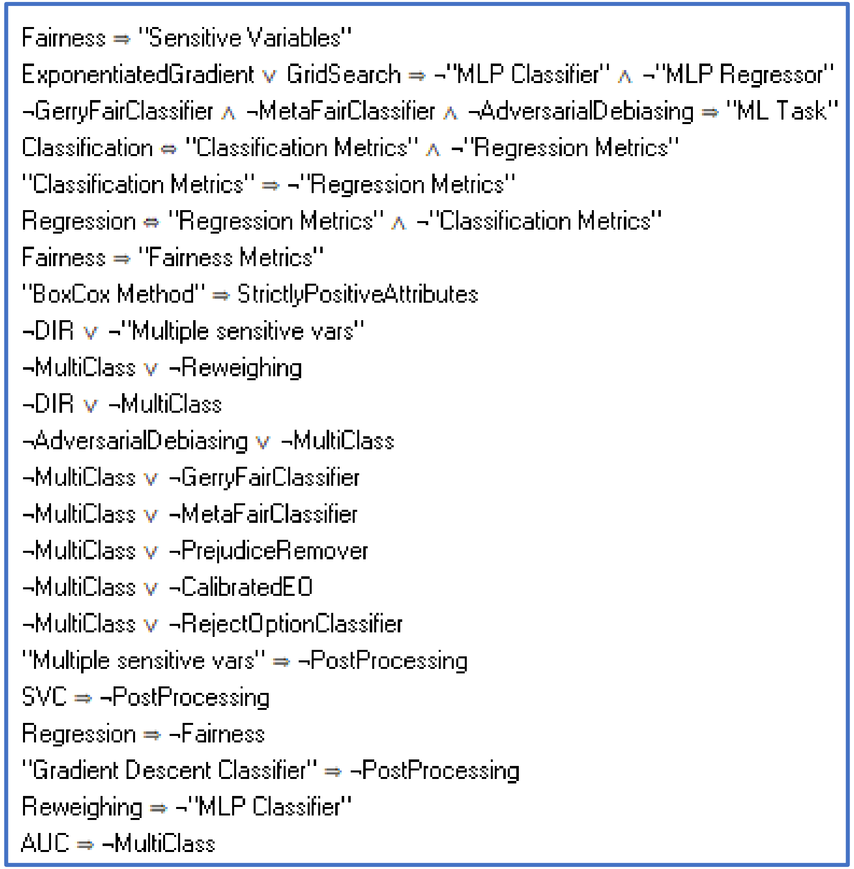
MANILA's Architecture

- MANILA uses the Extended Feature Model (ExtFM) formalism to model the evaluation workflow as a Software Product Line
- Eventually returns the setting with best fairness and effectiveness combination



MANILA’s ExtFM and Constraints

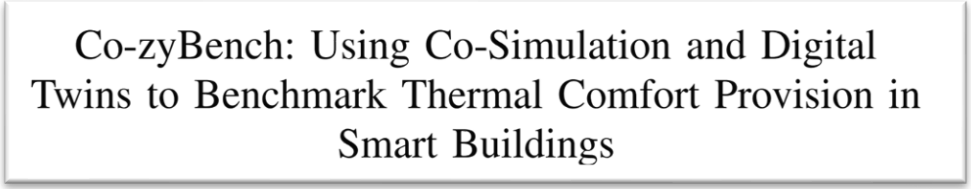




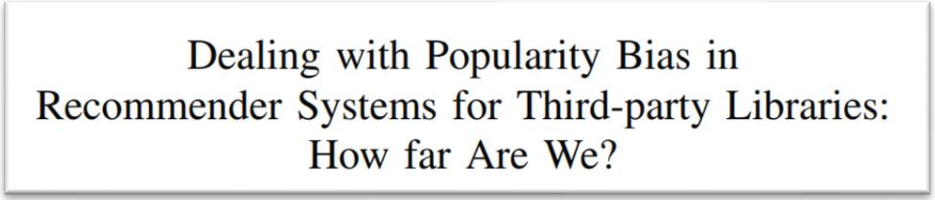
Fairness ⇒ "Sensitive Variables"
ExponentiatedGradient ∨ GridSearch ⇒ ¬"MLP Classifier" ∧ ¬"MLP Regressor"
¬GerryFairClassifier ∧ ¬MetaFairClassifier ∧ ¬AdversarialDebiasing ⇒ "ML Task"
Classification ⇔ "Classification Metrics" ∧ ¬"Regression Metrics"
"Classification Metrics" ⇒ ¬"Regression Metrics"
Regression ⇔ "Regression Metrics" ∧ ¬"Classification Metrics"
Fairness ⇒ "Fairness Metrics"
"BoxCox Method" ⇒ StrictlyPositiveAttributes
¬DIR ∨ ¬"Multiple sensitive vars"
¬MultiClass ∨ ¬Reweighing
¬DIR ∨ ¬MultiClass
¬AdversarialDebiasing ∨ ¬MultiClass
¬MultiClass ∨ ¬GerryFairClassifier
¬MultiClass ∨ ¬MetaFairClassifier
¬MultiClass ∨ ¬PrejudiceRemover
¬MultiClass ∨ ¬CalibratedEO
¬MultiClass ∨ ¬RejectOptionClassifier
"Multiple sensitive vars" ⇒ ¬PostProcessing
SVC ⇒ ¬PostProcessing
Regression ⇒ ¬Fairness
"Gradient Descent Classifier" ⇒ ¬PostProcessing
Reweighing ⇒ ¬"MLP Classifier"
AUC ⇒ ¬MultiClass

Challenge 3: Custom Fairness Assessment

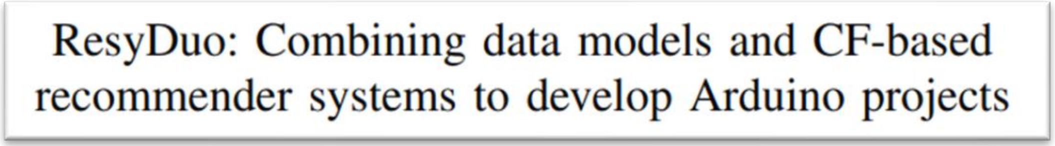
- Most of the fairness assessment tools available focus on specific definitions of fairness or cover traditional use cases
- What about non-traditional use cases (e.g., IoT or RecSys?)



Co-zyBench: Using Co-Simulation and Digital Twins to Benchmark Thermal Comfort Provision in Smart Buildings



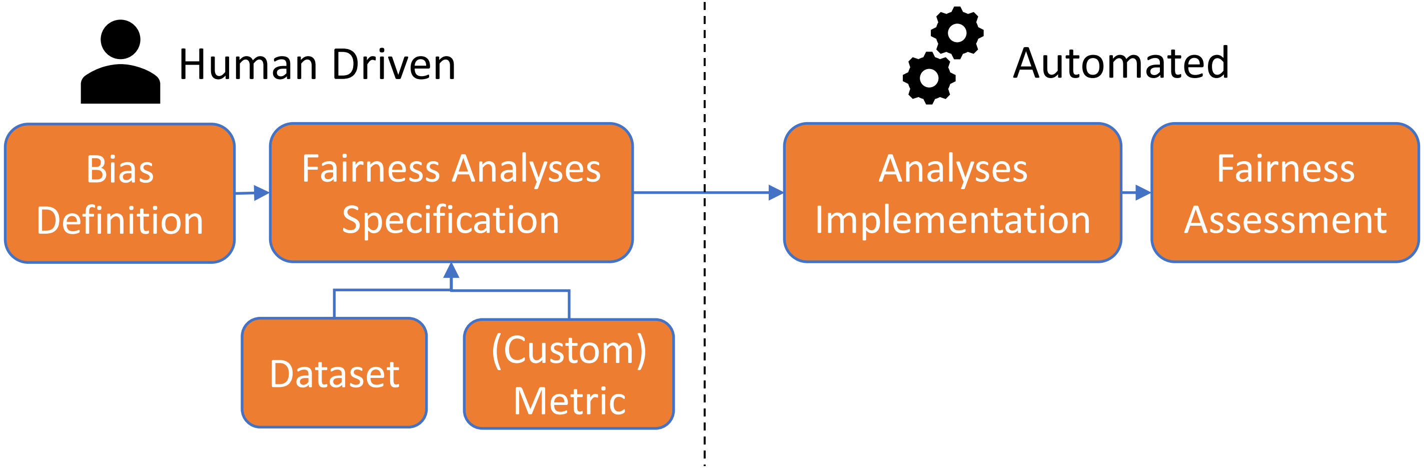
Dealing with Popularity Bias in Recommender Systems for Third-party Libraries: How far Are We?



ResyDuo: Combining data models and CF-based recommender systems to develop Arduino projects

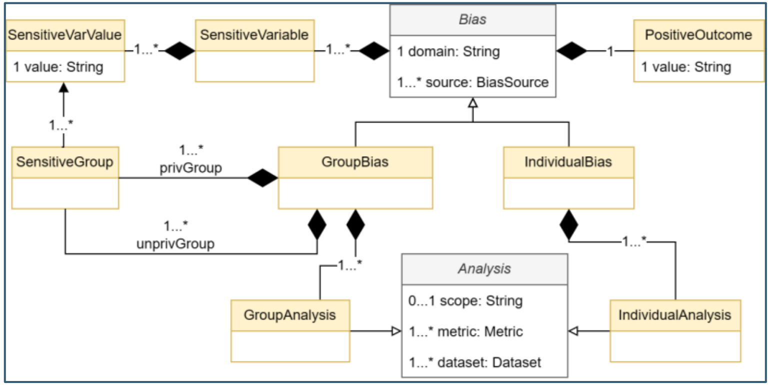
Addressing Challenge 3: MODNESS [4]

- MODNESS is a model-driven framework to design, implement and execute fairness analyses
- Allows to define fairness analyses from a high-level bias definition and create custom metrics

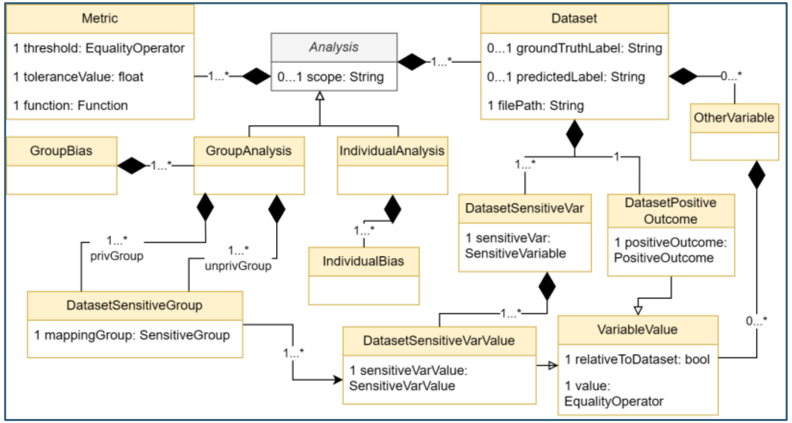


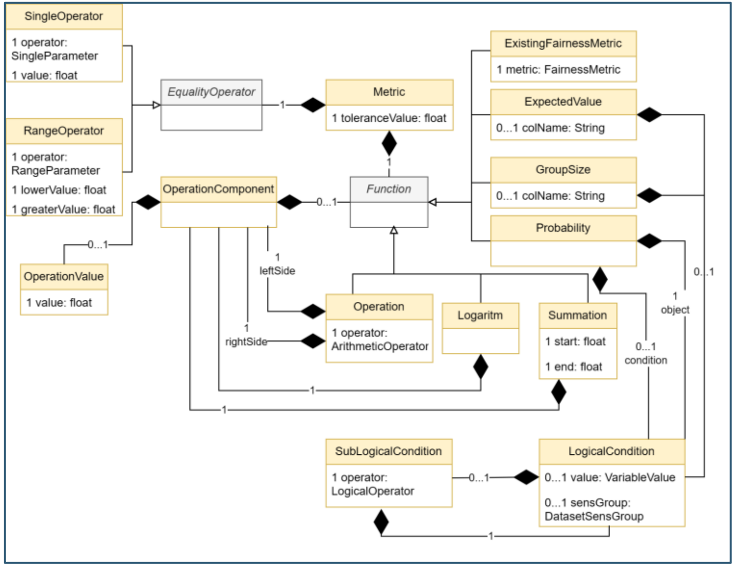
Bias and Fairness Metamodel

Bias Definition


SensitiveVarValue
1 value: String
SensitiveVariable
Bias
1 domain: String
1...* source: BiasSource
PositiveOutcome
1 value: String
SensitiveGroup
GroupBias
IndividualBias
Analysis
0...1 scope: String
1...* metric: Metric
1...* dataset: Dataset
GroupAnalysis
IndividualAnalysis

Fairness Analysis


Metric
1 threshold: EqualityOperator
1 toleranceValue: float
1 function: Function
Analysis
0...1 scope: String
Dataset
0...1 groundTruthLabel: String
0...1 predictedLabel: String
1 filePath: String
OtherVariable
GroupBias
GroupAnalysis
IndividualAnalysis
IndividualBias
DatasetSensitiveVar
1 sensitiveVar: SensitiveVariable
DatasetPositive Outcome
1 positiveOutcome: PositiveOutcome
DatasetSensitiveGroup
1 mappingGroup: SensitiveGroup
DatasetSensitiveVarValue
1 sensitiveVarValue: SensitiveVarValue
VariableValue
1 relativeToDataset: bool
1 value: EqualityOperator


SingleOperator
1 operator: SingleParameter
1 value: float
RangeOperator
1 operator: RangeParameter
1 lowerValue: float
1 greaterValue: float
EqualityOperator
Metric
1 toleranceValue: float
ExistingFairnessMetric
1 metric: FairnessMetric
ExpectedValue
0...1 colName: String
GroupSize
0...1 colName: String
Probability
OperationComponent
Function
OperationValue
1 value: float
Operation
1 operator: ArithmeticOperator
Logaritm
Summation
1 start: float
1 end: float
SubLogicalCondition
1 operator: LogicalOperator
LogicalCondition
0...1 value: VariableValue
0...1 sensGroup: DatasetSensGroup

Metric Definition

Many Challenges Are Still Open

- Addressing the trade-off between fairness and other quality properties (e.g., effectiveness or computational complexity)
- Early identification of features leading to bias in a dataset
- Recommender system for bias definitions and metrics from user requirement
- And many more...

References

DEMV:

- [1] G. d’Aloisio, G. Stilo, A. Di Marco, e A. D’Angelo, «Enhancing Fairness in Classification Tasks with Multiple Variables: A Data-and Model-Agnostic Approach», in International Workshop on Algorithmic Bias in Search and Recommendation, Springer, 2022.
- [2] G. d’Aloisio, A. D’Angelo, A. Di Marco, e G. Stilo, «Debiaser for Multiple Variables to enhance fairness in classification tasks», Information Processing & Management.

MANILA:

- [4] G. d’Aloisio, A. Di Marco, e G. Stilo, «Democratizing Quality-Based Machine Learning Development through Extended Feature Models», in FASE 2023.

MODNESS:

- [5] G. d'Aloisio, C. Di Sipio, A. Di Marco, and D. Di Ruscio. «How fair are we? From conceptualization to automated assessment of fairness definitions », arXiv preprint arXiv:2404.09919.

Thank you for your attention!

giordano.daloisio@graduate.univaq.it

UNIVERSITÀ
DEGLI STUDI
DELL'AQUILA

DISIM
Dipartimento di Ingegneria
e Scienze dell'Informazione
e Matematica