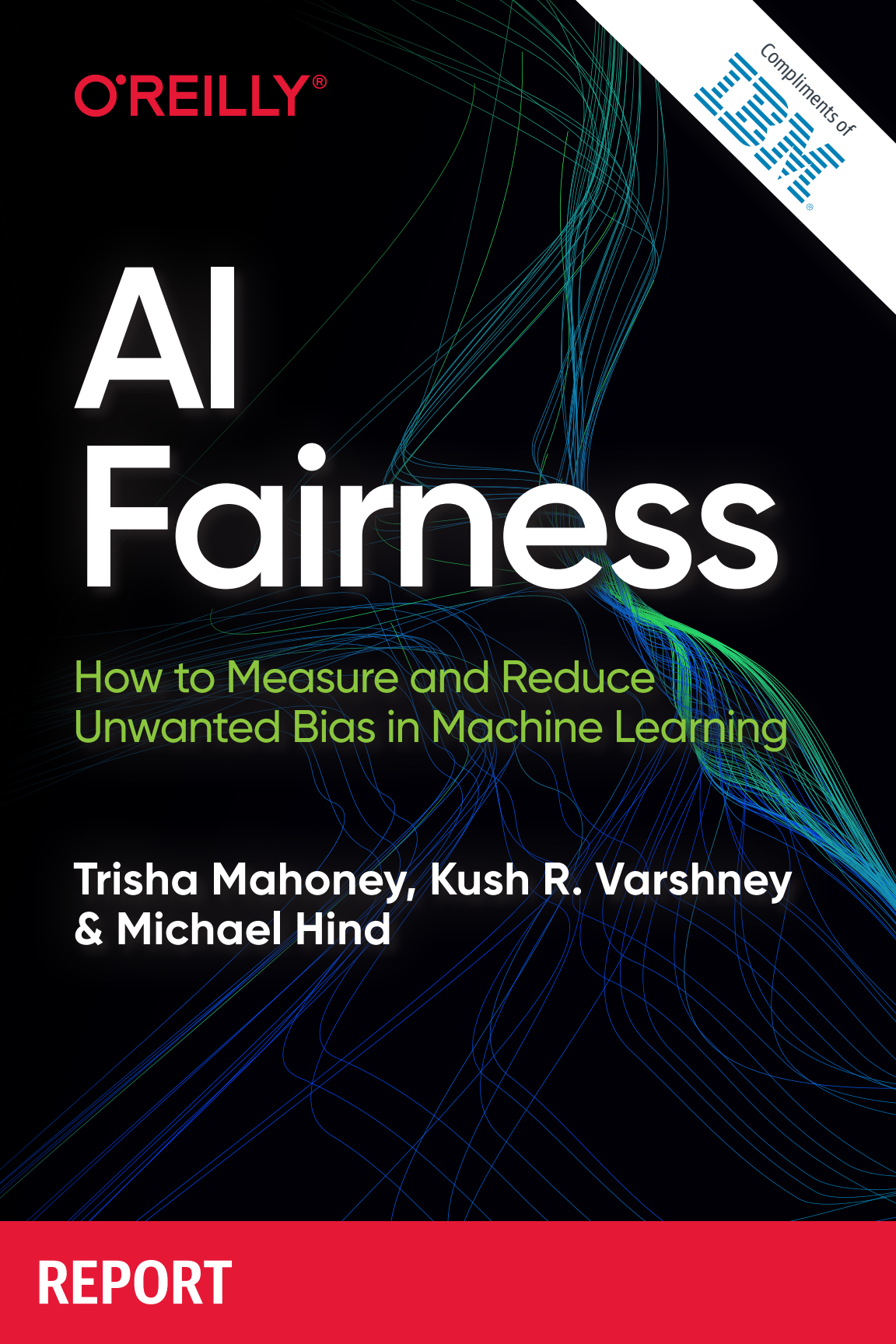
O'REILLY®
Compliments of
IBM
AI
Fairness
How to Measure and Reduce
Unwanted Bias in Machine Learning
Trisha Mahoney, Kush R. Varshney
& Michael Hind
REPORT

AI Fairness

How to Measure and Reduce Unwanted Bias in Machine Learning

Trisha Mahoney, Kush R. Varshney, and Michael Hind

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

AI Fairness
by Trisha Mahoney, Kush R. Varshney, and Michael Hind

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Jonathan Hassell
Development Editor: Sarah Grey
Production Editor: Nan Barber
Copyeditor: Octal Publishing, LLC

Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

March 2020: First Edition

Revision History for the First Edition
2020-02-25: First Release

This work is part of a collaboration between O'Reilly and IBM. See our statement of editorial independence.

978-1-492-07763-3

[LSI]

Table of Contents

Introduction

Are Human Decisions Less Biased Than Automated Ones?

This report takes data science leaders and practitioners through the key challenges of defining fairness and reducing unfair bias throughout the machine learning pipeline. It shows why data science teams need to engage early and authoritatively on building trusted artificial intelligence (AI). It also explains in plain English how organizations should think about AI fairness, as well as the trade-offs between model bias and model accuracy. Much has been written on the social injustice of AI bias, but this report focuses on how teams can mitigate unfair machine bias by using the open source tools available in AI Fairness 360 (AIF360).

Developing unbiased algorithms involves many stakeholders across a company. After reading this report, you will understand the many factors that you need to consider when defining fairness for your use case (such as legal compliance, ethics, and trust). You will also learn that there are several ways to define fairness, and thus many different ways to measure and reduce unfair bias. Although not all bias can be removed, you will learn that organizations should define acceptable thresholds for both model accuracy and bias.

In this report, we provide an overview of how AI bias could negatively affect every industry as the use of algorithms becomes more prevalent. Next, we discuss how to define fairness and the source of bias. Then, we discuss bias in the machine learning pipeline and the current bias mitigation toolkits. We then go into the AI Fairness 360 toolkit with a focus on how to measure bias and attempt to remove

it. We conclude with a Python tutorial of a credit-scoring use case and provide some guidance on fairness and bias.

AI Fairness Is Becoming Increasingly Critical

AI is increasingly being used in highly sensitive areas such as health care, hiring, and criminal justice, so there has been a wider focus on the implications of bias and unfairness embedded in it. We know that human decision-making in many areas is biased and shaped by our individual or societal biases, which are often unconscious. One may assume that using data to automate decisions would make everything fair, but we now know that this is not the case. AI bias can come in through societal bias embedded in training datasets, decisions made during the machine learning development process, and complex feedback loops that arise when a machine learning model is deployed in the real world.

Extensive evidence has shown that AI can embed human and societal biases and deploy them at scale. Many experts are now saying that unwanted bias might be the major barrier that prevents AI from reaching its full potential. One such case that has received media attention is that of Correctional Offender Management Profiling for Alternative Sanctions (COMPAS), an algorithm used to predict whether defendants in Broward County, Florida, among other jurisdictions, were likely to commit a crime if they were not remanded. A 2016 investigation by journalists at ProPublica found that the COMPAS algorithm incorrectly labeled African-American defendants as "high-risk" at nearly twice the rate it mislabeled white defendants. This illustrates the significant negative impact an AI algorithm can have on society. So how do we ensure that automated decisions are less biased than human decision-making?

Defining Fairness

Fairness is a complex and multifaceted concept that depends on context and culture. Defining it for an organization's use case can thus be difficult. There are at least 21 mathematical definitions of fairness.[1] These are not just theoretical differences in how to

1 Arvind Narayanan, "21 Fairness Definitions and Their Politics," tutorial at Conference on Fairness, Accountability, and Transparency, February 2018. https://oreil.ly/MhDrk.

measure fairness; different definitions focus on different aspects of fairness and thus can produce entirely different outcomes. Fairness researchers have shown that it is impossible to satisfy different definitions of fairness at the same time.[2] The University of Chicago has created a decision tree that is useful in thinking through how organizations can define fairness.

Many definitions focus either on individual fairness (treating similar individuals similarly) or on group fairness (making the model's predictions/outcomes equitable across groups). Individual fairness seeks to ensure that statistical measures of outcomes are equal for similar individuals. Group fairness partitions a population into groups defined by protected attributes and seeks to ensure that statistical measures of outcomes are equal across groups.

Within those who focus on group fairness, there are two opposing worldviews. These can be roughly summarized as "We're All Equal" (WAE) and "What You See is What You Get" (WYSIWYG).[3] The WAE worldview holds that all groups have the same abilities, while the WYSIWYG worldview holds that observations reflect the abilities of groups.

For example, let's consider how universities could define group fairness using SAT scores as a feature for predicting success in college. The WYSIWYG worldview would say that the score correlates well with future success and can be used to compare the abilities of applicants accurately. In contrast, the WAE worldview would say that SAT scores may contain structural biases, so its distribution being different across groups should not be mistaken for a difference in distribution of ability.

In addition, different bias-handling algorithms address different parts of the machine learning life cycle; understanding how, when, and why to use each research contribution is challenging even for experts in algorithmic fairness. As a result, AI stakeholders and practitioners need clarity on how to proceed. Currently the burden is on the practitioners, who face questions such as these:

2 J. Kleinberg et al., "Inherent Tradeoffs in the Fair Determination of Risk Scores," in *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*, January 2017, 43.1–43.23.

3 S. A. Friedler et al., "On the (Im)possibility of Fairness," September 2016, *arXiv: 1609.07236*. https://oreil.ly/ps4so.

- Should our data be debiased?
- Should we create new classifiers that learn unbiased models?
- Is it better to correct predictions from the model?

This report helps you to strategize on how to approach such questions for your organization's specific use case.

Where Does Bias Come From?

Algorithmic bias is often discussed in machine learning, but in most cases the underlying data, rather than the algorithm, is the main source of bias. Consider supervised learning, which is the most common form of machine learning. Its goal is to find a mathematical function that takes data points of numerical, ordinal, or categorical features as inputs and predicts correct labels for those data points. Models are trained on data, and often that data contains human decisions that reflect the effects of societal or historical inequities.

For example, in a credit scoring model, the features might be income, education level, and occupation of an applicant, and the label might be whether an applicant defaults three years later. The algorithm finds the desired function by training on a large set of already labeled examples. Although the function fits the training data, when it is applied to new data points, it must generalize to predict well. In its most basic form, fitting is performed to optimize a criterion such as average accuracy.

The biggest problem with machine learning models is that the training distribution does not always match the desired distribution. If the present reality puts certain individuals at a systematic disadvantage, the training data distribution is likely to reproduce that disadvantage rather than reflecting a fairer future. Biases such as those against African-Americans in the criminal justice system and women in employment can be present whenever judges and hiring managers make decisions. These decisions are reflected in the training data and subsequently baked into future machine learning model decisions. Some types of bias that can be introduced into data include the following:

Sample bias
: Sample bias occurs when one population is overrepresented or underrepresented in a training dataset. An example of this would be a recruiting tool that has been predominantly trained on white male job candidates.

Label bias
: Label bias occurs when the annotation process introduces bias during the creation of training data. For example, the people labeling the data might not represent a diverse group of locations, ethnicities, languages, ages, and genders, and can bring their implicit personal biases into their labels. This can lead to labels that are skewed in ways that yield systematic disadvantages to certain groups.

Outcome proxy bias
: Outcome proxy bias occurs when the machine learning task is not specified appropriately. For example, if one would like to predict the likelihood of a person committing a crime, using arrests as a proxy is biased because arrest rates are greater in neighborhoods with more police patrols. Also, being arrested does not imply guilt. Similarly, using the cost of a person to a health system is a biased proxy for the person's quality of health.

Bias and Machine Learning

Machine learning models make predictions of an outcome for a particular instance. For example, given an instance of a loan application, a model might predict whether an applicant will repay the loan. The model makes these predictions based on a training dataset for which many other instances (other loan applications) and actual outcomes (whether the borrowers repaid) are provided. Thus, a machine learning algorithm will attempt to find patterns, or *generalizations*, in the training dataset to use when a prediction for a new instance is needed. For example, one pattern might be that if a person has a salary greater than $40,000 and outstanding debt less than $5,000, they will repay the loan. In many domains, this technique, called *supervised machine learning*, has worked very well.

However, sometimes the patterns such models find are undesirable or even illegal. For example, our loan repayment model might determine that age plays a significant role in the prediction of repayment, because the training dataset happened to have better repayment

rates for one age group than for another. This raises two problems. First, the training dataset might not be representative of the true population of people of all age groups. Second, even if it is representative, it is illegal to base a decision on an applicant's age, regardless of whether this is a good prediction based on historical data.

AIF 360 addresses this problem with *fairness metrics* and *bias mitigators*. We can use fairness metrics to check for bias in machine learning workflows. We can use bias mitigators to overcome bias in the workflow to produce a fairer outcome. The loan scenario describes an intuitive example of illegal bias. However, not all undesirable bias in machine learning is illegal; bias can also manifest in subtler ways. For example, a loan company might want a diverse portfolio of customers across all income levels and thus will deem it undesirable to make more loans to high-income customers than to low-income customers. Although doing so is not illegal or unethical, it is undesirable for the company's strategy.

Can't I Just Remove Protected Attributes?

When building machine learning models, many data scientists assume that they can just remove protected attributes (i.e., race, gender, age) to avoid unfair bias. However, there are many features that are too closely correlated to protected attributes, which makes it easy to reconstruct a protected attribute such as race even if you drop it from your training set.

An example of this is when Amazon rolled out its Prime Free Same-Day Delivery service for many products on orders over $35. Eleven months after rolling out the program, Amazon offered same-day service in 27 metropolitan areas. However, a 2016 Bloomberg News analysis found that six of the major cities serviced were excluding predominantly black zip codes to varying degrees. And in Atlanta, Chicago, Dallas, Washington, Boston, and New York, black citizens were about half as likely as white residents to live in neighborhoods with access to Amazon same-day delivery. How did this occur? Amazon focused its same-day service model on zip codes with a high concentration of Prime members, not on race. Yet in cities where most of those paying members are concentrated in predominantly white parts of town, looking at numbers instead of people did not prevent a data-driven calculation from reinforcing long-entrenched inequality in access to retail services.

Conclusion

Bias mitigation is not a silver bullet. Fairness is a multifaceted, context-dependent social construct that defies simple definition. The metrics and algorithms in AIF 360 can be viewed through the lens of distributive justice and do not capture the full scope of fairness in all situations. The toolkit should be used in only a very limited setting: allocation or risk assessment problems with well-defined protected attributes in which one would like to have some sort of statistical or mathematical notion of sameness. Even then, the code and collateral contained in AIF 360 are only a starting point for a broader discussion among multiple stakeholders on overall decision-making workflows.

CHAPTER 1

Understanding and Measuring Bias with AIF 360

Bias can occur at any stage in the machine learning pipeline (Figure 1-1), and fairness metrics and bias mitigation algorithms can be performed at various stages within the pipeline. We recommend checking for bias as often as possible, using as many metrics as are relevant to your application. We also recommend integrating continuous bias detection into your automated pipeline. AIF360 is compatible with the end-to-end machine learning workflow and is designed to be easy to use and extensible. Practitioners can go from raw data to a fair model easily while comprehending the intermediate results, and researchers can contribute new functionality with minimal effort.

In this chapter, we look at current tools and terminology and then begin looking at how AIF360's metrics work.

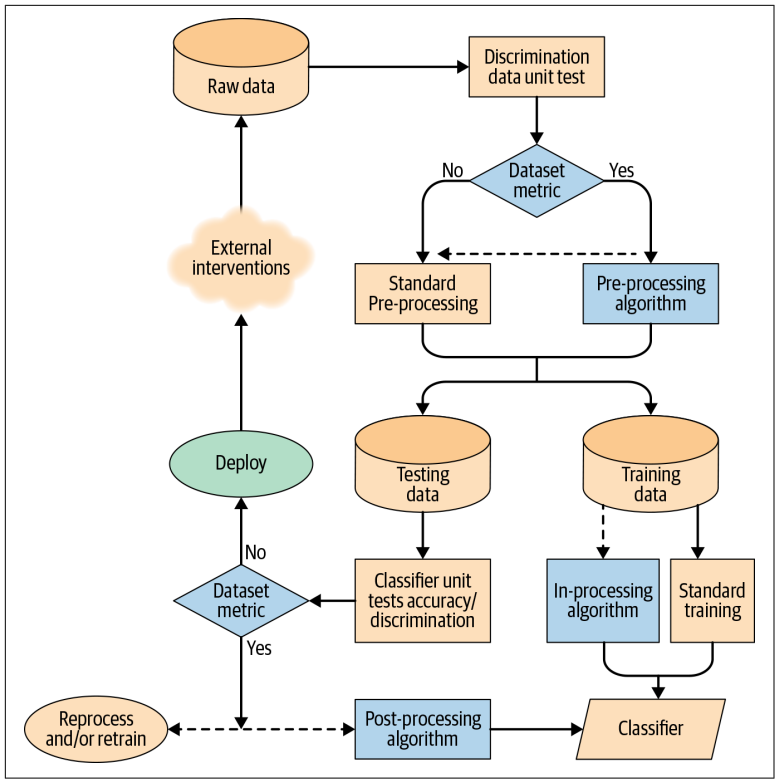

Figure 1-1. Bias in the machine learning pipeline

Tools and Terminology

Several open source libraries have been developed in recent years to contribute to building fairer AI models. Most address only bias detection, not mitigating bias. Just a handful of toolkits (like Themis-ML and Fairness Comparison) address both, but they are often limited for commercial use due to their usability and license restrictions. IBM fairness researchers took on the initiative to unify these efforts, as shown in Table 1-1, and bring together a comprehensive set of bias metrics, bias mitigation algorithms, bias metric explanations, and industrial usability in one open source toolkit with AIF360.

Table 1-1. Other open source libraries on bias and fairness

Open source library	Advantages/disadvantages
Fairness Measures	Provides several fairness metrics, including difference of means, disparate impact, and odds ratio. It also provides datasets, but some are not in the public domain and require explicit permission from the owners to access or use the data.
FairML	Provides an auditing tool for predictive models by quantifying the relative effects of various inputs on a model's predictions, which can be used to assess the model's fairness.
FairTest	Checks for associations between predicted labels and protected attributes. The methodology also provides a way to identify regions of the input space where an algorithm might incur unusually high errors. This toolkit also includes a rich catalog of datasets.
Aequitas	This is an auditing toolkit for data scientists as well as policy makers; it has a Python library and website where data can be uploaded for bias analysis. It offers several fairness metrics, including demographic, statistical parity, and disparate impact, along with a "fairness tree" to help users identify the correct metric to use for their particular situation. Aequitas's license does not allow commercial use.
Themis	An open source bias toolbox that automatically generates test suites to measure discrimination in decisions made by a predictive system.
Themis-ML	Provides fairness metrics, such as mean difference, some bias mitigation algorithms,[a] additive counterfactually fair estimator,[b] and reject option classification.[c] The repository contains a subset of the methods described in this book.
Fairness Comparison	Includes several bias detection metrics as well as bias mitigation methods, including disparate impact remover,[d] prejudice remover,[e] and two-Naive Bayes.[f] Written primarily as a test bed to allow different bias metrics and algorithms to be compared in a consistent way, it also allows additional algorithms and datasets.

[a] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," in *Knowledge and Information Systems* 33, no. 1, 2012, 1–33.

[b] M. J. Kusner et al., "Counterfactual Fairness," in *Advances in Neural Information Processing Systems*, 2017, 4,069–4,079.

[c] F. Kamiran et al., "Decision Theory for Discrimination-Aware Classification," in *Proceedings of IEEE International Conference on Data Mining*, 2012, 924–929.

[d] M. Feldman et al., "Certifying and Removing Disparate Impact," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 2015, 259–268.

[e] T. Kamishima et al., "Fairness-Aware Classifier with Prejudice Remover Regularizer," in *Machine Learning and Knowledge Discovery in Databases*, 2012, 35–50.

[f] T. Calders and S. Verwer, "Three Naive Bayes Approaches for Discrimination-Free Classification," in *Data Mining and Knowledge Discovery* 21, 2010, 77–292.

AIF360 is an extensible, open source toolkit for measuring, understanding, and reducing AI bias. It combines the top bias metrics, bias mitigation algorithms, and metric explainers from fairness

researchers across industry and academia. The goals of the AIF360 toolkit are as follows:

- To promote a deeper understanding of fairness metrics and mitigation techniques
- To enable an open common platform for fairness researchers and industry practitioners to share and benchmark their algorithms
- To help facilitate the transition of fairness research algorithms for use in an industrial setting

AIF360 currently includes Python packages that implement techniques from eight published papers across the greater AI fairness research community. There are currently 77 bias detection metrics and 10 bias mitigation algorithms that can be called in a standard way similar to scikit-learn's fit/transform/predict paradigm. The toolkit is open source and contains documentation, demonstrations, and other artifacts.

Terminology

To use the AIF360 toolkit effectively, it helps to be familiar with the relevant fairness terminology. Here are a few of the specialized machine learning terms to know:

Favorable label
: A label whose value corresponds to an outcome that provides an advantage to the recipient (such as receiving a loan, being hired for a job, not being arrested)

Protected attribute
: An attribute that partitions a population into groups whose outcomes should have parity (such as race, gender, caste, and religion)

Privileged value (of a protected attribute)
: A protected attribute value indicating a group that has historically been at a systemic advantage

Fairness metric
: A quantification of unwanted bias in training data or models

Discrimination/unwanted bias
: Although *bias* can refer to any form of preference, fair or unfair, our focus is on *undesirable bias* or *discrimination*, which is when specific privileged groups are placed at a systematic advantage and specific unprivileged groups are placed at a systematic disadvantage. This relates to attributes such as race, gender, age, and sexual orientation.

Which Metrics Should You Use?

AIF360 currently contains 77 fairness metrics to measure bias (see Figure 1-2), so it can be difficult to understand which metrics to use for each use case. Because there is no one best metric for every case, it is recommended to use several metrics, carefully chosen with the guidance of subject matter experts and key stakeholders within your organization. AIF360 has both difference and ratio versions of metrics that essentially convey the same information, so you can choose based on the comfort of the users examining the results.

AIF360's API documentation allows you to view all of its metrics and algorithms along with a definition of each metric, algorithm, or explainer; the research on which it is based; its source code; and its parameters and exceptions. There are several classes of metrics listed in the API categories that follow. If your use case requires metrics on training data, use the ones in the `DatasetMetric` class (and in its child classes, such as `BinaryLabelDatasetMetric`). If the application requires metrics on models, use the `ClassificationMetric` class. Here are the relevant classes:

BinaryLabelDatasetMetric
: A class for computing metrics based on a single binary label dataset

ClassificationMetric
: A class for computing metrics based on a two binary label dataset

Sample Distortion Metric
: A class for computing metrics based on two structured datasets

Utility Functions
: A helper script for implementing metrics

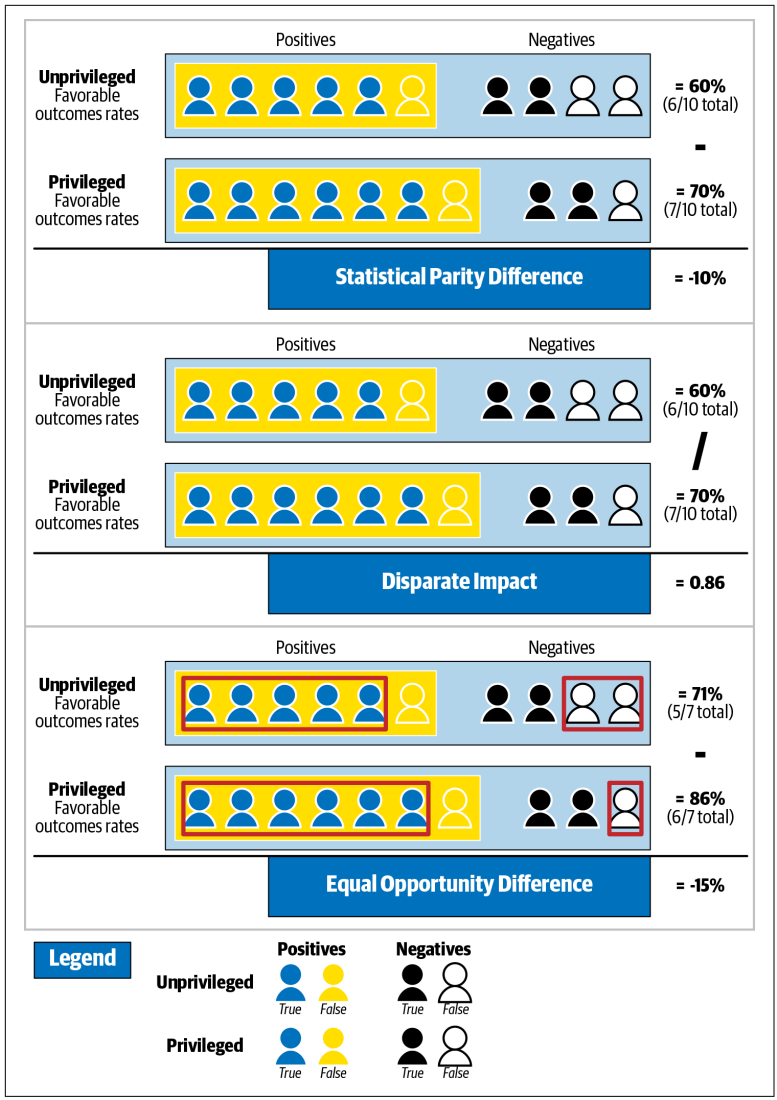

Figure 1-2. How bias is measured

Individual Versus Group Fairness Metrics

If your use case is focused on individual fairness, use the metrics in the `SampleDistortionMetric` class or the consistency metric. If your focus is on group fairness, use the `DatasetMetric` class (and its child classes, such as the `BinaryLabelDatasetMetric`) as well as `ClassificationMetric`. And if your use case is concerned with *both*

individual and group fairness and requires the use of a single metric, use the generalized entropy index and its specializations to the Theil index and the coefficient of variation in the `ClassificationMetric` class. However, multiple metrics, including ones from both individual and group fairness, can be examined simultaneously.

Worldviews and Metrics

If your use case follows the WAE worldview, use the demographic parity metrics (such as `disparate_impact` and `statistical_parity_difference`). If your use case follows the WYSIWYG worldview, use the equality of odds metrics (such as `average_odds_difference` and `average_abs_odds_difference`). Other group fairness metrics (often labeled "equality of opportunity") lie in between the two worldviews. Relevant metrics include the following:

- `false_negative_rate_ratio`
- `false_negative_rate_difference`
- `false_positive_rate_ratio`
- `false_positive_rate_difference`
- `false_discovery_rate_ratio`
- `false_discovery_rate_difference`
- `false_omission_rate_ratio`
- `false_omission_rate_difference`
- `error_rate_ratio`
- `error_rate_difference`

Dataset Class

The `Dataset` class and its subclasses are a key abstraction that handle all forms of data. Training data is used to learn classifiers, whereas testing data is used to make predictions and compare metrics. Besides these standard aspects of a machine learning pipeline, fairness applications also require associating protected attributes with each instance or record in the data. To maintain a common format, independent of what algorithm or metric is being applied, AIF360 structures the `Dataset` class so that all relevant attributes (features, labels, protected attributes, and their respective

identifiers) are present in and accessible from each instance of the class. Subclasses add further attributes that differentiate the dataset and dictate with which algorithms and metrics it can interact.

Structured data is the primary type of dataset studied in fairness literature and is represented by the `StructuredDataset` class. Further distinction is made for a `BinaryLabelDataset`—a structured dataset that has a single label per instance that can take only one of two values: favorable or unfavorable. Unstructured data can be accommodated in the AIF360 architecture by constructing a parallel class to the `StructuredDataset` class, without affecting existing classes or algorithms that do not apply to unstructured data.

Transparency in Bias Metrics

We now turn our focus to the broader issue of improving transparency in bias metrics.

Explainer Class

The AIF 360 toolkit prioritizes the need to explain fairness metrics, so IBM researchers developed an `Explainer` class to coincide with the `Metrics` class that provides further insights about computed fairness metrics. Different subclasses of varying complexity that extend the `Explainer` class can be created to output explanations that are meaningful to different kinds of users. The explainer capability implemented in the first release of AIF360 is basic reporting through printed JSON outputs. Future releases might include fine-grained localization of bias, actionable recourse analysis, and counterfactual fairness.

AI FactSheets

One barrier to fairness and trust in AI is the lack of standard practices to document how AI services are created, tested, trained, deployed, and evaluated. Organizations are asking for transparency in AI services as well as how they should and should not be used. To address this need, AI FactSheets[1] provide information about a

1 M. Arnold et al., "FactSheets: Increasing Trust in AI Services through Supplier's Declarations of Conformity" in *IBM Journal of Research and Development* 63, no. 4/5, (July/September 2019): 6, https://oreil.ly/Bvo10.

machine learning model's important characteristics in a similar way to food nutrition labels or appliance specification sheets. They would contain information such as system operation, training data, underlying algorithms, test setup and results, performance benchmarks, fairness and robustness checks, intended uses, maintenance, and retraining. Example questions could be as follows:

- Does the dataset used to train the service have a datasheet or data statement?
- Were the dataset and model checked for biases? If so, describe the bias policies that were checked, bias checking methods, and results.
- Was any bias mitigation performed on the dataset? If so, describe the mitigation method.
- Are algorithm outputs explainable or interpretable? If so, explain how (for example, through a directly explainable algorithm, local explainability, or explanations via examples).
- Who is the target user of the explanation (machine learning expert, domain expert, general consumer, regulator, etc.)?
- Was the service tested on any additional datasets? Do they have a datasheet or data statement?
- Describe the testing methodology.
- Is usage data from service operations retained?
- What behavior is expected if the input deviates from training or testing data?
- How is the overall workflow of data to the AI service tracked?

This information will aid users in understanding how the service works, its strengths and limitations, and whether it is appropriate for the application they are considering.

CHAPTER 2

Algorithms for Bias Mitigation

We can measure data and model fairness at different points in the machine learning pipeline. In this chapter, we look at the pre-processing, in-processing, and post-processing categories of bias mitigation algorithms.

Most Bias Starts with Your Data

AIF360's bias mitigation algorithms are categorized based on where in the machine learning pipeline they are deployed, as illustrated in Figure 2-1. As a general guideline, you can use its pre-processing algorithms if you can modify the training data. You can use in-processing algorithms if you can change the learning procedure for a machine learning model. If you need to treat the learned model as a black box and cannot modify the training data or learning algorithm, you will need to use the post-processing algorithms.

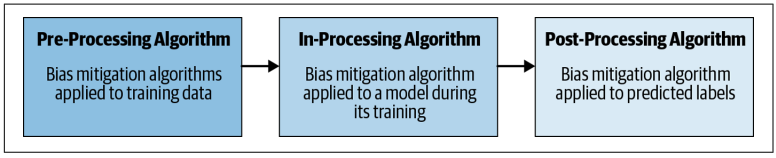

Figure 2-1. Where can you intervene in the pipeline?

Pre-Processing Algorithms

Pre-processing is the optimal time to mitigate bias given that most bias is intrinsic to the data. With pre-processing algorithms, you attempt to reduce bias by manipulating the training data before training the algorithm. Although this is conceptually simple, there are two key issues to consider. First, data can be biased in complex ways, so it is difficult for an algorithm to translate one dataset to a new dataset which is both accurate and unbiased. Second, there can be legal issues involved: in some cases, training the decision algorithm on nonraw data can violate antidiscrimination laws.

The following are the pre-processing algorithms in AIF360 as of early 2020:

Reweighing Pre-Processing
: Generates weights for the training samples in each (group, label) combination differently to ensure fairness before classification. It does not change any feature or label values, so this is ideal if you are unable to make value changes.

Optimized Pre-Processing
: Learns a probabilistic transformation that edits the features and labels in the data with group fairness, individual distortion, and data fidelity constraints and objectives.

Learning Fair Representations
: Finds a latent representation that encodes the data well but obfuscates information about protected attributes.

Disparate-Impact Remover
: Edits feature values to increase group fairness while preserving rank ordering within groups.

If your application requires transparency about the transformation, Disparate-Impact Remover and Optimized Pre-Processing are ideal.

In-Processing Algorithms

You can incorporate fairness into the training algorithm itself by using in-processing algorithms that penalize unwanted bias. In-processing is done through fairness constraints (e.g., older people should be accepted at the same rate as young people) that influence

the loss function in the training algorithm. Here are some of the most widely used in-processing algorithms in AIF360:

Adversarial Debiasing
: Learns a classifier to maximize prediction accuracy and simultaneously reduces an adversary's ability to determine the protected attribute from the predictions. This approach leads to a fair classifier because the predictions can't carry any group discrimination information that the adversary can exploit.

Prejudice Remover
: Adds a discrimination-aware regularization term to the learning objective.

Meta Fair Classifier
: Takes the fairness metric as part of the input and returns a classifier optimized for the metric.

Post-Processing Algorithms

In some instances, a user might have access to only black-box models, so post-processing algorithms must be used. AIF360 post-processing algorithms (see Figure 2-2) can be used to reduce bias by manipulating the output predictions after training the algorithm. Post-processing algorithms are easy to apply to existing classifiers without retraining. However, a key challenge in post-processing is finding techniques that reduce bias and maintain model accuracy. AIF360's post-processing algorithms include the following:

Equalized Odds
: Solves a linear program to find probabilities with which to change output labels to optimize equalized odds.

Calibrated Equalized Odds
: Optimizes over calibrated classifier score outputs to find probabilities with which to change output labels with an equalized odds objective.

Reject Option Classification
: Gives favorable outcomes to unprivileged groups and unfavorable outcomes to privileged groups in a confidence band around the decision boundary with the highest uncertainty.

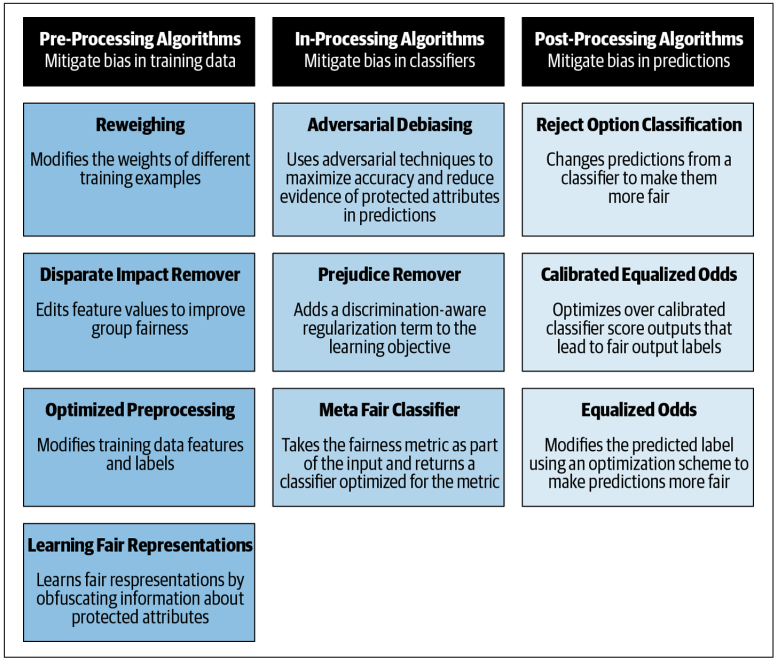

Figure 2-2. Algorithms in the machine learning pipeline

Continuous Pipeline Measurement

The probability distribution governing data can change over time, resulting in the training data distribution drifting away from the actual data distribution. It can also be difficult to obtain sufficient labeled training data to model the current data distribution correctly. This is known as *dataset drift* or *dataset shift*. Training models on mismatched data can severely degrade prediction accuracy and performance, so bias measurement and mitigation should be integrated into your continuous pipeline measurements, in a similar manner that functional testing is integrated with each application change.

CHAPTER 3
Python Tutorial

This chapter introduces the basic functionality of AIF 360 to interested developers who might not have a background in bias detection and mitigation. It offers a Python tutorial on detecting and mitigating racial bias through the example of income estimation using the Optimized Pre-Processing algorithm. You can also find this tutorial on GitHub, where more sophisticated machine learning workflows are given in the author tutorials and demonstration notebooks.[1]

As these two examples illustrate, a bias detection and/or mitigation toolkit needs to be tailored to the particular bias of interest. More specifically, it needs to know the attribute or attributes, called *protected attributes*, that are of interest: race is one example of a *protected attribute*; age is a second.

We will use the Adult Census Income dataset, splitting it into training and test datasets. We will look for bias in the creation of our machine learning model, which is intended to help decide whether an individual's annual income should be set greater than $50,000, based on various personal attributes. The protected attribute will be race, with "1" (white) and "0" (not white) being the values for the privileged and unprivileged groups, respectively. For this first tutorial, we check for bias in the initial training data, mitigate the bias, and recheck.

1 This demonstration is very similar to the Credit Scoring Tutorial. It is meant as an alternative introduction using a different dataset and mitigation algorithm.

Here are the steps involved:

1. Write import statements.
2. Set bias detection options, load dataset, and split between train and test.
3. Compute fairness metric on original training dataset.
4. Mitigate bias by transforming the original dataset.
5. Compute fairness metric on transformed training dataset.

Step 1: Import Statements

As with any Python program, the first step is to import the necessary packages. In the code that follows, we import several components from the AIF360 package. We import a custom version of the Adult dataset with certain continuous features sorted into categories, metrics to check for bias, and classes related to the algorithm we use to mitigate bias. We also import some other useful packages:

```
import sys
sys.path.append("../")

import numpy as np

from aif360.metrics import BinaryLabelDatasetMetric

from aif360.algorithms.preprocessing.optim_preproc import \
    OptimPreproc
from aif360.algorithms.preprocessing.optim_preproc_helpers. \
    data_preproc_functions import load_preproc_data_adult
from aif360.algorithms.preprocessing.optim_preproc_helpers. \
    distortion_functions import get_distortion_adult
from aif360.algorithms.preprocessing.optim_preproc_helpers. \
    opt_tools import OptTools

from IPython.display import Markdown, display
In [2]:
np.random.seed(1)
```

Step 2: Load Dataset, Specify Protected Attribute, and Split Dataset into Train and Test

In step 2 we load the initial dataset, setting the protected attribute to be race. We then split the original dataset into training and testing datasets. Although we use only the training dataset in this tutorial, a normal workflow would also use a test dataset for assessing the efficacy (accuracy, fairness, etc.) during the development of a machine learning model. Finally, we set two variables (to be used in step 3) for the privileged (`1`) and unprivileged (`0`) values for the race attribute. These are key inputs for detecting and mitigating bias, which will be step 3 and step 4:

```
dataset_orig = load_preproc_data_adult(['race'])

dataset_orig_train, dataset_orig_test = \
    dataset_orig.split([0.7], shuffle=True)

privileged_groups = [{'race': 1}] # White
unprivileged_groups = [{'race': 0}] # Not white
```

Step 3: Compute Fairness Metric on Original Training Dataset

Now that we've identified the protected attribute, `race`, and defined privileged and unprivileged values, we can use AIF360 to detect bias in the dataset. One simple test is to compare the percentage of favorable results for the privileged and unprivileged groups, subtracting the former percentage from the latter. A negative value indicates less favorable outcomes for the unprivileged groups. This is implemented in the method called `mean_difference` on the `BinaryLabelDatasetMetric` class. The code below performs this check and displays the output:

```
metric_orig_train = BinaryLabelDatasetMetric(dataset_orig_train,
             unprivileged_groups=unprivileged_groups,
             privileged_groups=privileged_groups)
display(Markdown("#### Original training dataset"))
print("Difference in mean outcomes between unprivileged and \
    privileged groups = %f" % \
    metric_orig_train.mean_difference())
    Original training dataset
```

```
Difference in mean outcomes between unprivileged and \
    privileged groups = -0.104553
```

Step 4: Mitigate Bias by Transforming the Original Dataset

The previous step showed that the privileged group was getting 10.5% more positive outcomes in the training dataset. Because this is not desirable, we are going to try to mitigate this bias in the training dataset. As stated earlier, this is called *pre-processing mitigation* because it happens before the creation of the model.

We choose the Optimized Pre-Processing algorithm,[2] which is implemented in the `OptimPreproc` class in the *aif360.algorithms.preprocessing* directory. This algorithm transforms the dataset to have more equity in positive outcomes on the protected attribute for the privileged and unprivileged groups.

The algorithm requires some tuning parameters, which are set in the `optim_options` variable and passed as an argument along with some other parameters, including the two variables containing the unprivileged and privileged groups defined in step 3.

We then call the fit and transform methods to perform the transformation, producing a newly transformed training dataset (`dataset_transf_train`). Finally, we ensure alignment of features between the transformed and the original dataset to enable comparisons:

```
optim_options = {
"distortion_fun": get_distortion_adult,
"epsilon": 0.05,
"clist": [0.99, 1.99, 2.99],
"dlist": [.1, 0.05, 0]
}
OP = OptimPreproc(OptTools, optim_options)

OP = OP.fit(dataset_orig_train)
dataset_transf_train = OP.transform(dataset_orig_train, \
    transform_Y=True)

dataset_transf_train = dataset_orig_train.align_datasets( \
```

2 F. Calmon et al., "Optimized Pre-Processing for Discrimination Prevention," in *Advances in Neural Information Processing Systems*, 2017.

```
    dataset_transf_train)
Optimized Preprocessing: Objective converged to 0.000000
```

Step 5: Compute Fairness Metric on Transformed Dataset

Now that we have a transformed dataset, we can check how effective our efforts have been in removing bias by using the same metric we used for the original training dataset in step 3. Again, we use the function `mean_difference` in the `BinaryLabelDatasetMetric` class:

```
metric_transf_train = BinaryLabelDatasetMetric( \
    dataset_transf_train, unprivileged_groups= \
    unprivileged_groups, privileged_groups=privileged_groups)
display(Markdown("#### Transformed training dataset"))
print("Difference in mean outcomes between unprivileged and \
    privileged groups = %f" % metric_transf_train. \
    mean_difference())
Transformed training dataset
Difference in mean outcomes between unprivileged and \
    privileged groups = -0.051074
```

We see the mitigation step was very effective: the difference in mean outcomes is now –0.051074. So, we went from a 10.5% advantage for the privileged group to a 5.1% advantage for the privileged group—a reduction by more than half!

A more complete use case would take the next step and see how the transformed dataset affects the accuracy and fairness of a trained model. This is implemented in the demonstration notebook in the examples directory of the AIF360 toolkit, called *demo_reweighing_preproc.ipynb*.

CHAPTER 4
Conclusion

The Future of Fairness in AI

Lack of trust in machine learning may be the single greatest factor that prevents AI from reaching its full potential. To build this trust, data scientists must consider measures beyond predictive accuracy. Mitigating bias and ensuring fairness, along with providing robustness and transparency, are essential to ushering in a world in which machine learning systems are partners that humanity can count on.

Fairness is difficult to define for a given application domain and is not something that data scientists should be tackling alone. The inclusion of multiple stakeholders and perspectives is essential before any modeling is undertaken. After some level of consensus is reached on the desirable values and goals for an AI system, the tools provided in AIF 360 can be utilized to detect, understand, and mitigate unwanted bias. These tools address only a narrow sliver of overall fairness issues, and they should be used in conjunction with procedural and organizational measures.

About the Authors

Trisha Mahoney is an AI tech evangelist for IBM with a focus on fairness and bias. Trisha has spent the last 10 years working on artificial intelligence and Cloud solutions at several Bay Area tech firms including Salesforce, IBM, and Cisco. Prior to that, Trisha spent eight years working as a data scientist in the chemical detection space. She holds an electrical engineering degree and an MBA in Technology Management.

Kush R. Varshney is a principal research staff member and manager with IBM Research AI, where he leads the Machine Learning group in the Foundations of Trusted AI department. He is the founding codirector of the IBM Science for Social Good initiative. He applies data science and predictive analytics to human capital management, healthcare, olfaction, computational creativity, public affairs, international development, and algorithmic fairness. Kush holds an SM in electrical and computer engineering with honors from Cornell and an SM and PhD in electrical engineering and computer science from MIT.

Michael Hind is a distinguished research staff member with IBM Research AI. Michael's team has successfully transferred technology to various parts of IBM and launched several successful open source projects, such as AI Fairness 360 and AI Explainability 360. His work on the Jikes RVM open source project was recognized with the SIGPLAN Software Award in 2012. Michael is an ACM Distinguished Scientist, and a member of IBM's Academy of Technology and holds a PhD in computer science from NYU.