Master's Thesis

# Towards Detection and Mitigation of Dataset Bias through Adaptive Reweighting


**Student:**

**Gabriel Hayat**
**14-821-060**
**ghayat@student.ethz.ch**
**ghayat@mit.edu**

**Supervisors:**

**David Lindner**
**Prof. Andreas Krause**
**Department of Computer Science**
**Learning & Adaptive Systems Group**
**ETH Zurich**

**Schrasing Tong**
**Prof. Lalana Kagal**
**Computer Science and Artificial Intelligence Lab**
**Decentralized Information Group**
**Massachusetts Institute of Technology**



Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

## Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

---

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

**Title of work** (in block letters):

TOWARDS DETECTION AND MITIGATION OF DATASET BIAS THROUGH ADAPTIVE REWEIGHTING

**Authored by** (in block letters):

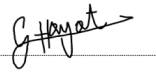
*For papers written by groups the names of all authors are required.*

| **Name(s):** | **First name(s):** |
|---|---|
| HAYAT | GABRIEL |
| | |
| | |
| | |
| | |

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

| **Place, date** | **Signature(s)** |
|---|---|
| ZURICH, 24/08/2021 |  |
| | |
| | |
| | |
| | |

*For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.*

# Acknowledgments

This thesis concludes my Computer Science M.Sc. program from ETH Zurich that has been filled with enriching and fun experiences. I had the chance to accomplish this six-month project in the *Decentralized Information Group* at the Massachusetts Institute of Technology. Unfortunately, due to COVID-19, I was unable to perform my research onsite. However, the wonderful people I've met through this journey made the experience as exciting as if I were there in person. Ph.D. student Schrasing Tong has followed this project very closely from day one. Through the countless remote meetings, he has always supported me with his never-ending innovative ideas. Thank you ! Furthermore, I'd like to thank Prof. Lalana Kagal for her guidance during this work and for making the partnership with MIT possible. In addition, I would like to mention the amazing services that the university offers, such as the Satori cluster, that allowed extensive and thorough experiments.

From the side of my home university ETH Zurich, I have been closely supervised by Prof. Andreas Krause's *Learning & Adaptive Systems Group*. More specifically, I would like to thank Ph.D. student David Lindner for his mentorship as well as for the many constructive feedback regarding this document.

I want to thank my family for their contagious positive energy and their enthusiasm. Lastly, I would like to dedicate this thesis to my grandfather with whom I have had many inspiring discussions on the subject.

# Abstract

The growing field of fair machine learning aims to ensure that decisions guided by algorithms are fair towards every demographic group of the population. Datasets often contain biases which unfairly disadvantage certain groups, and classifiers trained on such datasets can inherit these unfair inclinations. These have two major causes: (i) training data bias where there is consistent discrimination towards certain groups of the training set and (ii) overfitting bias, which occurs through severe imbalances in the representation of these groups in the dataset. As a first step, this thesis presents two algorithms that seek to correct dataset bias by reweighting the data points without changing the labels. Our procedures are general purpose approaches that can be used with many learning algorithms as well as enforce most of the prevalent fairness definitions. We evaluate our approaches on the *Adult* dataset [30]. In a second step, we assess our proposed algorithms in the context of *Computer Vision*. Despite significant advancements in the fairness world, there is a need for a framework to impose fairness when the protected attributes are not explicitly stated. We propose a methodology to infer bias from a given dataset and combine this approach with our mitigation algorithms to provide a complete pipeline for *unsupervised fairness*.

# Table of contents


**1 Introduction** **8**
- 1.1 Motivation . . . 8
- 1.2 Outline of the Thesis . . . 10

**2 Foundations** **12**
- 2.1 Logistic Regression . . . 12
- 2.2 Residual Neural Network . . . 13
- 2.3 Principal Component Analysis . . . 15
- 2.4 $k$-means clustering . . . 16

**3 Related Work** **17**
- 3.1 Fairness definitions . . . 17
  - 3.1.1 Notation . . . 17
  - 3.1.2 Demographic parity . . . 17
  - 3.1.3 Equalized odds . . . 18
  - 3.1.4 Equality of opportunity . . . 18
  - 3.1.5 Individual Fairness . . . 19
  - 3.1.6 Disparate Treatment vs Disparate Impact . . . 19
- 3.2 Bias detection . . . 21
  - 3.2.1 Representation Learning . . . 21
  - 3.2.2 Clustering . . . 22
- 3.3 Bias mitigation . . . 24
  - 3.3.1 Preprocessing . . . 24
  - 3.3.2 Optimization at training time . . . 25
  - 3.3.3 Post-processing . . . 26
  - 3.3.4 Federated Learning . . . 28

**4 Method** **29**
- 4.1 Cluster reweighting . . . 29
- 4.2 Sample reweighting . . . 32
- 4.3 Clustering images for bias detection . . . 34

**5 Experiments with Tabular data** **37**
- 5.1 Adult Data Set . . . 37
- 5.2 Comparative models . . . 38
  - 5.2.1 Vanilla . . . 38
  - 5.2.2 Reweighting preprocessing . . . 38
  - 5.2.3 Adversarial debiasing in-processing . . . 39

5.2.4 Receiver Operating Characteristic post-processing . . . . . 40
5.3 Experiments . . . . . 42
5.4 Towards Subgroup Fairness . . . . . 44
5.5 Base rate Analysis . . . . . 46

**6 Experiments with Image data** **50**
6.1 Sport Dataset . . . . . 50
6.2 Experiments . . . . . 51
6.2.1 Clustering . . . . . 51
6.2.2 Model evaluation . . . . . 52
6.3 Adjusting the Bias . . . . . 53
6.4 Semi-supervised Clustering . . . . . 55

**7 Conclusion** **57**
7.1 Contributions and directions for future work . . . . . 57
7.2 Final remarks . . . . . 58

**8 Appendix** **63**
8.1 Appendix A: the Adult dataset . . . . . 63
8.2 Appendix B: the Sport dataset . . . . . 64
8.2.1 Sample examples . . . . . 64
8.2.2 Sample preprocessing . . . . . 65
8.3 Appendix C: Experimental settings . . . . . 65
8.3.1 Tabular dataset . . . . . 65
8.3.2 Image dataset . . . . . 65
8.4 Appendix D: UTKFace dataset . . . . . 66
8.5 Appendix E: Description of code packages . . . . . 66

# List of Figures


1 An example of bias in an object detection task . . . . . 9
2 Shortcut connections used in Residual Neural Network . . . . . 13
3 Investigating the sample reweighting model's misclassified data points 48
4 Sample examples from the Sport dataset . . . . . 64

# List of Tables


1 Experimental results evaluated on a gender-split tabular dataset . . . 42
2 Experimental results evaluating subgroup fairness. . . . . 44
3 Experimental results of the base rate models on tabular datasets . . . 47
4 Experimental results on an image dataset split in terms of jersey color 51
5 Evaluating the ability of our approach to expose bias on different image datasets . . . . . 53
6 Semi-supervised vs Unsupervised Clustering . . . . . 55

# 1 Introduction

## 1.1 Motivation

How should one program an intelligent system to adopt *good* behavior ? When it comes to well-defined industrial automation tasks, the performance goals are often easily expressed. An intelligent agent will determine its behavior following the aim of maximizing an utility function and choose the actions that will offer the best possible consequences toward meeting its goals. The risk of an utilitarian agent, which seeks to maximize the utility of its task, is to fall into the hypothetical scenario of the *paperclip maximizer* (Nick Bostrom, 2003): we ask a robot to make as many paperclips as possible, and to achieve this objective, ends up absorbing all human resources and causing the end of humanity. This illustrates the existential risk that a general artificial intelligence agent may pose to humanity when programmed to pursue even seemingly harmless goals, and the necessity of incorporating machine ethics into the design of artificial intelligent systems.

When making automated decisions that affect people's livelihood and well-being, we must make sure that these systems make ethical decisions. These considerations include a broad range of concerns about fairness, trust, accountability, transparency, and privacy [44]. Human decision makers are also faced by these issues while making similar judgements. However, as machine learning systems are human-developed software, we expect to understand and approve all of their actions as well as have control over the consequences of their deployment. Nevertheless, this is rarely the case as these systems often lack transparency and their downstream impact is often unpredictable. Moreover, because of their scalability, poor decisions made by machine learning systems can have severe consequences on our society [23].

Undesirable effects of intelligent agents have already been seen in the social context. Indeed, many areas have recently been affected by harmful bias. Examples include criminal justice. A decision support tool called *Correctional Offender Management Profiling for Alternative Sanction* (COMPAS, [46]) is used in U.S. courts to estimate the recidivism risk of defendants for decisions on pre-trial detention. Some sources such as ProPublica claimed that COMPAS produced results biased against people of color, as they received systematically worse scores as compared to white individuals [48]. Another example is by the Google Vision Cloud, a computer vision service, that produces image labeling. It was shown that the system gave a different labeling depending on the skin color of the person represented in the image, causing Google to publicly apologize [37]. Figure 1 illustrates an example of that bias. It shows a thermometer labeled as a gun when held by a dark-skinned hand, while it is labeled

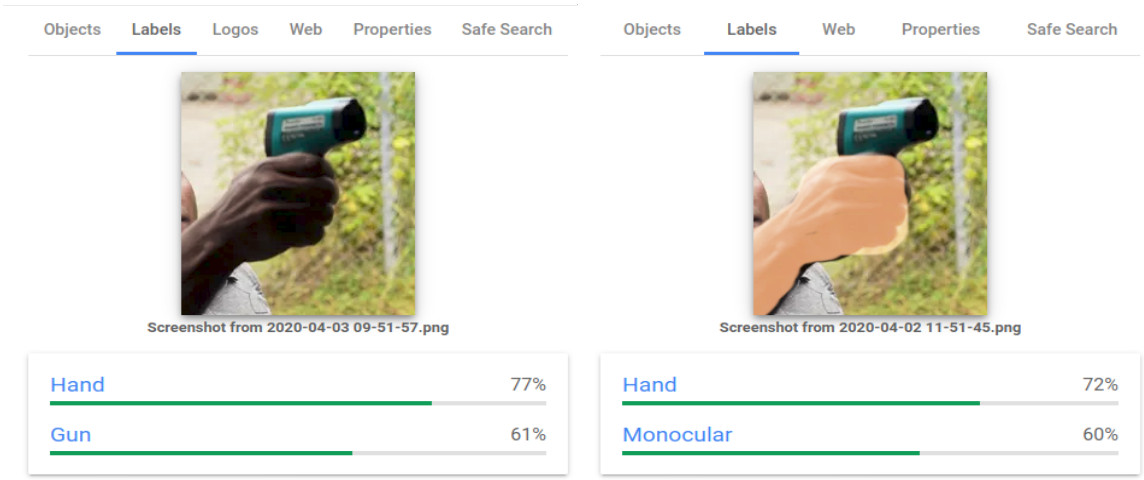



Figure 1: Image labeling bias [37]

as a monocular when adding a salmon-color overlay over the hand.

These examples illustrate the potential of algorithmic discrimination in our society. In this thesis, we will examine how the representation of sub-populations in a dataset influence the disparities in model predictions and suggest algorithms to mitigate this bias. Furthermore, we will examine fairness in a computer vision setting, where these issues become more important every day. More generally, the hope is to expand the horizons of readers to think deeply about fairness issues while developing algorithms that will affect individuals' daily lives. With the recent advancement in AI, it is essential that researchers take this factor into account when designing tomorrow's intelligent systems.

## 1.2 Outline of the Thesis

Fairness has become a major concern in the field of Artificial Intelligence, in particular in Computer Vision, where this technology is being used today in a wide variety of real-world applications. These include for instance social media (e.g. advanced image query in Facebook), 3D model building (e.g. creation of 3D building models from aerial images used in systems such as Google Maps [9]) or face detection (e.g. for improved digital camera focusing [42]). In many cases, classification models are trained on historical datasets, some of which have shown biases against certain groups. In this case, the model leverages this bias to improve upon its training accuracy and if not cautious, will reproduce this unwanted behavior in the future. This leads to undesirable effects such as the two examples previously stated.

Although the research community has made significant advances in algorithmic fairness when handling tabular data (e.g. [1, 4, 5, 8, 12]), fairness in the image domain has not yet received the same attention. Some of the tasks are unsupervised [[41, 45, 50]], i.e. the attributes and labels are not always clearly defined, which impedes any fairness evaluation. In this case, imposing some notion of fairness first requires the model to recognize and cluster together data with similar properties, and identify the presence of sensitive information in the image. In this work, we seek to develop a pipeline that is able to identify potential sources of harmful bias as well as mitigate its effect when training a machine learning model on the data.

This thesis tackles this problem by adopting a bottom-up approach, gradually adding components to eventually reach our objective. Section 2 gives a brief overview of some common models and algorithms used in the literature, namely *Logistic Regression*, *Residual Neural network*, *Principal Component analysis* and *k-means clustering*. Readers already familiar with these concepts are welcome to skip this chapter.

In Section 3, we review some of the existing research in the field of fairness. More specifically, we first take a critical look at common statistical fairness criteria. Although these definitions are widely accepted in the literature, one must be cautious when using them to prove the fairness of their algorithm. Indeed, satisfying a few definitions is rarely enough to guarantee a discrimination-free algorithm. Next, we discuss some of the recent advances in two distinct areas: *bias detection* and *bias mitigation*. The former attempts to identify sources of bias in large datasets, i.e is equal treatment achieved across all represented sub-populations ? The latter lists some of the fairness-aware machine learning algorithms that already exist, distinguishing between three common approaches: preprocessing, in-processing, and post-processing methods.

Subsequently, in Section 4, we present the main contributions of this thesis. Given a specific clustering of the dataset, the first algorithm iteratively re-weights the clusters as to enforce some fairness definition while maintaining a good overall accuracy. That is, each cluster, represented by a set of *similar* samples, is given a weight that is adjusted through the training process as to satisfy some fairness constraint. The second algorithm works with similar mechanics, except that the weights are attributed to samples instead of clusters. Lastly, we introduce a procedure that enables the detection of bias within an image dataset.

Section 5 focuses on evaluating our algorithms on a tabular dataset, namely the *Adult dataset* [30]. After having briefly introduced the dataset, we list some fairness-aware algorithms that we use as comparisons to our models as well as explain their mechanics. After evaluating the performances of the listed algorithms, we discuss two relevant extensions, namely: *subgroup fairness* and *base rate analysis*. Advocates of subgroup fairness argue that imposing fairness across different groups is not enough; it should be enforced over an exponentially or infinitely large collection of sub-populations. In the base rate analysis subsection, we discuss how the classification rate of each individual sub-population affects the overall rate when merging them together.

In addition, Section 6 investigates the performances of our algorithms on an Image dataset. This section focuses on an issue that often exists when dealing with images: how can fairness be properly addressed when the sensitive attributes are not clearly labelled ? We make use of every tool presented in the thesis in order to provide an adequate solution in response to this question. Furthermore, we explore the limitations of our methods by manually altering the level of bias in the dataset. Lastly, we develop a *semi-supervised* approach to better partition the dataset in terms of sensitive attributes.

Finally, Section 7 first summarizes the contributions of the thesis. We then draw our conclusions and highlight directions for future work.

# 2 Foundations

In this chapter, we introduce some background material. We briefly describe different models and techniques that will prove to be useful at a later stage of this thesis.

## 2.1 Logistic Regression

*Logistic Regression* is a statistical analysis method in which there is one or more independent variables that determine an outcome. It is used to model the probability of a binary dependent variable. An output with more than two values are modeled by a *multinomial logistic regression* (not relevant for this paper, the interested reader can see [29]). In the logistic model, the *log-odds*, i.e. the logarithm of the odds, is a linear combination of one or more independent variables (i.e. the predictors). The model's independent variables can be binary (i.e. indicator functions) or continuous and outputs the sample's probability of having a positive label.

Consider a dataset $\mathcal{X} \in \mathbb{R}^{n \times d}$ with sample $\mathbf{x} \in \mathbb{R}^d$ and the binary (Bernoulli) response variable $\mathbf{Y} \in \{0, 1\}$, we denote $p = \mathbb{P}(\mathbf{Y} = 1)$. We assume a linear relationship between the sample's features and the log-odds (also called *logit*) of event $\mathbf{Y} = 1$:

$$
\begin{aligned}
ln(\frac{p}{1-p}) = \mathbf{x}^T\beta \quad \Longleftrightarrow \quad p &= \frac{e^{\mathbf{x}^T\beta}}{e^{\mathbf{x}^T\beta} + 1} \\
&= \frac{1}{1 + e^{-\mathbf{x}^T\beta}} \\
&= S(\mathbf{x}^T\beta)
\end{aligned}
$$

$\beta \in \mathbb{R}^d$ represents the model's parameters. The function $S$ that converts log-odds to probability is defined as the *logistic function*. Although logistic regression simply models probability of event $\mathbf{Y} = 1$ in terms of its inputs, it can easily be interpreted as a classifier by choosing a cutoff value $t \in [0, 1]$, where classifying inputs with probability lower or greater than $t$ is classified as the negative or positive label respectively. In Section 5, we make use of logistic regression as a classification model to differentiate between two outcome classes .

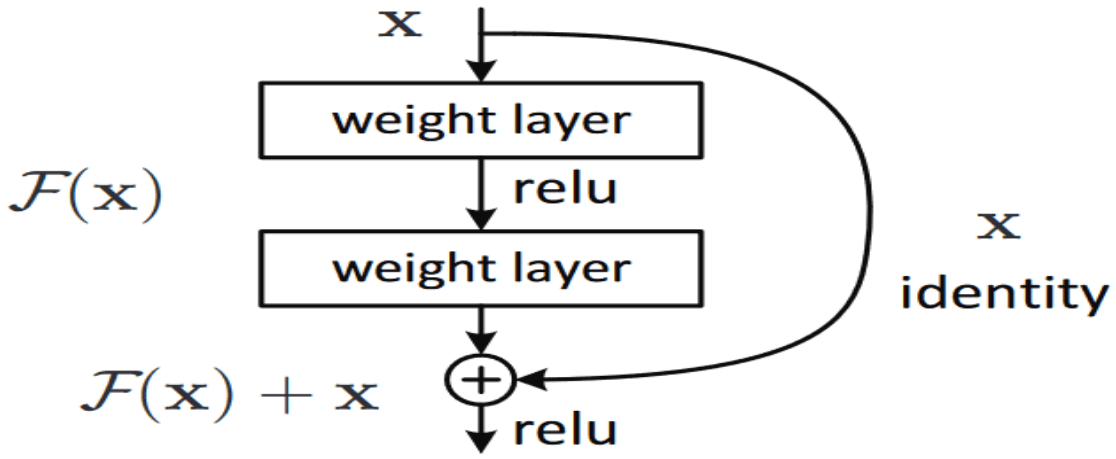



Figure 2: Shortcut connections used in Residual Neural Network [19]

## 2.2 Residual Neural Network

A *Residual Neural Network* (Resnet) is an Artificial Neural Network that adresses two common problems of Deep Neural Networks, namely *vanishing/exploding gradients* and the *accuracy degradation* problem. The former is encountered when training artificial neural networks with gradient-based learning methods and backpropagation. In some cases, due to the depth of the network, the gradient of a particular weight of one of the layers will end up being vanishingly small, effectively preventing the weight from changing its value. In the worst case, this may completely stop the neural network from further training. Similarly, the problem of exploding gradient is encountered when the gradients can take arbitrarily large values. Secondly, the problem of accuracy degradation highlights the fact that increasing the network's depth can sometimes lead to accuracy saturation. Given that a "shallow" model is able to capture the task at hand, we expect its deeper counterpart to obtain at least the same accuracy. However, when the model gets deeper, it becomes increasingly difficult for the layers to propagate the information from shallow layers leading to the loss of information. This phenomena is verified by [32], where they show that the training error of a 20-layer network is lower than the one of a 56-layer network on multiple datasets.

The authors of [32] address the two above problems by the use of *shortcut connections* or *residual connections*, that are used to skip one or more layers. The shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers (see Figure 2). In the context of the vanishing gradient problem, these connections allow information from the earlier parts of the network to

be passed to the deeper parts of the network, helping to maintain signal propagation even in deeper networks. With regards to the accuracy degradation problem, [32] hypothesizes that this problem is partly due to the fact that combining multiple non-linear layers might result in difficulties in approximating identity mappings. Thus, using more layers than necessary can deteriorate the (training) accuracy. When using shortcut connections, identity mappings can easily be reproduced by driving the weights of the multiple nonlinear layers toward zero.

For the rest of the thesis, we choose to use a Resnet architecture when dealing with images. In the case of binary classification, we replace the last layer by a dense layer with two output neurons. We use a pretrained model from Pytorch's torchvision library, freeze all the layers except from the last two which are trained on our dataset. We refer the reader to [32] for the details of the model's architecture.

## 2.3 Principal Component Analysis

Principal component analysis (PCA) is the process of computing the principal components and using them to perform a change of basis on the data. It is commonly used for dimensionality reduction by projecting each data point onto the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. The $i$-th principal component can be seen as a direction orthogonal to the first *i-1* principal components that maximizes the variance of the projected data. The PCA process involves multiple steps for which we give an overview in the rest of the section.

The first step involves performing *standardization* on the data. The aim of this step is that each variable contributes equally to the analysis of the data. If omitted, the variables with larger ranges will dominate over those with small ranges, which will lead to biased results. Once the data is standardized, one needs to compute the *correlation matrix* $\Sigma \in \mathbb{R}^{d \times d}$ (symmetric), where $d$ is the number of variables in the data. Recall that the *Spectral Theorem* states that any matrix is diagonalizable by an orthogonal matrix if and only if it is symmetric. Hence, the covariance matrix can be diagonalized:

$$\begin{aligned} &\Sigma = \mathbf{U}\Lambda\mathbf{U}^T \\ &\Lambda = diag(\lambda_1, \ldots, \lambda_d) \quad \lambda_1 \geq \cdots \geq \lambda_d \geq 0 \\ &\mathbf{U} = (\ \mathbf{u_1}\ \mathbf{u_2}\ \ldots\ \mathbf{u_d}\ ) \qquad \mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I} \end{aligned}$$

In this decomposition, $\lambda_i$ is an eigenvalue of the matrix $\Sigma$ and corresponds to the eigenvector $\mathbf{u}_i$, i.e. $\Sigma\mathbf{u_i} = \lambda_i\mathbf{u}_i$. Note that all eignenvalues are non-negative as the covariance matrix is semi-positive definite. $\mathbf{U}$ is an orthogonal matrix (unit length, orthogonal columns) and forms an eigenvector basis of $\Sigma$. The *principal components* that result from the decomposition are uncorrelated and most of the information within the initial variables is compressed into the first components.

In this thesis, PCA will be used in order to perform dimension reduction. Suppose we want to reduce the dimension of (normalized) data $\mathbf{X} \in \mathbb{R}^{d \times n}$ to $m$ dimensions, we construct the reduction matrix $\tilde{\mathbf{U}}$ by selecting the first $m$ eigenvectors of $\mathbf{U}$ and apply it to the original data:

$$\begin{aligned} &\tilde{\mathbf{U}} = (\ \mathbf{u_1}\ \mathbf{u_2}\ \ldots\ \mathbf{u_m}\ ) \quad m \leq d \\ &\mathbf{Z} = \tilde{\mathbf{U}}^T\,\mathbf{X} \in \mathbb{R}^{m \times n} \end{aligned}$$

Further analysis and processing can now be done on the smaller dataset $\mathbf{Z}$.

## 2.4 *k*-means clustering

$k$-means clustering is one of the simplest and popular unsupervised machine learning algorithms. It is a method of vector quantization, originally from signal processing, that aims to partition $N$ observations into $K$ clusters in which each observation belongs to the cluster with the nearest mean (i.e. cluster *centroid*), serving as a representative of the cluster. A cluster refers to a collection of data points aggregated together due to certain similarities. In other words, the $k$-means algorithm identifies $K$ centroids ($K$ is defined by the user), and then allocates every data point to the nearest cluster, while keeping the in-cluster sum of squares minimal.

Given a set of data points $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n} \in \mathbb{R}^d$, the task is to assign each data point to one of the $K$ clusters, i.e. find a function $\pi : \mathbb{R}^d \longrightarrow \{1, 2, \ldots, K\}$. Each cluster $j$ is represented by its centroid $\mathbf{u}_j \in \mathbb{R}^d$ and the mapping is induced via minimizing the Euclidean distance:

$$\pi(\mathbf{x}) = \arg\min_{j=1\ldots K} ||\mathbf{u}_j - \mathbf{x}||$$

We formalize the clustering problem as an optimization problem, i.e. find the centroids $\mathbf{u}_j$ and assignment function $\pi$ that minimize an objective. We encode $\pi$ via the indicator matrix $\mathbf{Z} \in \mathbb{R}^{N \times K}$, where:

$$z_{ij} = \begin{cases} 1 & \text{if } \pi(\mathbf{x}_i) = j \\ 0 & otherwise \end{cases} \qquad \sum_{j=1}^{K} z_{ij} = 1 \quad \forall i \in \{1, \ldots N\}$$

The first equality makes $\mathbf{Z}$ a sparse matrix, while the second enforces *hard clustering*, i.e. any data point $\mathbf{x}$ is assigned to exactly one cluster. We can now express the $k$-means objective as:

$$\begin{aligned} J(\mathbf{U}, \mathbf{Z}) &= \sum_{i=1}^{N} \sum_{j=1}^{K} z_{ij} ||\mathbf{x}_i - \mathbf{u}_j||^2 \qquad \mathbf{X} = [\ \mathbf{x}_1\ \ldots\ \mathbf{x}_n] \in \mathbb{R}^{d \times N} \text{ the data matrix} \\ &= ||\mathbf{X} - \mathbf{U}\mathbf{Z}^T||_{\mathbf{F}}^2 \qquad\qquad \mathbf{U} = [\ \mathbf{u}_1\ \ldots\ \mathbf{u}_K] \in \mathbb{R}^{d \times K} \text{ the centroid matrix} \end{aligned}$$

where $|| \cdot ||_F$ is the *Frobenius Norm* [1]. The objective $J$ is often minimized through an alternative minimization approach composed of two steps; (i) determining optimal centroids given assignment matrix $\mathbf{Z}$; (ii) determining optimal assignments given centroid matrix $\mathbf{U}$.

[1] The Frobenius Norm is defined as $||\mathbf{A}||_F = \sqrt{Tr(\mathbf{A}\mathbf{A}^T)}$

# 3 Related Work

Fairness research has grown in recent years. Since algorithms started impacting our daily lives, ensuring fair treatment of individuals has become a major concern. Concretely, statistical measures have been developed to model group and individual fairness (Section 3.1). Furthermore, these definitions can be enforced at different stages of the algorithm development, namely: preprocessing, post-processing and optimization at training time (Section 3.3). This section introduces the common definitions used to measure algorithm fairness as well as discusses the recent progress in the field.

## 3.1 Fairness definitions

Advancements in training machine learning systems to output fair decisions have defined two main notions of fairness in decision making: *group fairness* and *individual fairness*. Group fairness partitions a population into groups defined by *protected attributes* and seeks for some statistical measure to be equal across groups. These can be imposed as constraints or incorporated into a loss function in order to mitigate disproportional outcomes in the system's output predictions regarding a protected demographic [1]. On the other hand, individual fairness focuses on the individuals and seeks for similar entities to be treated similarly [3]. Here, we give the definitions of the most prevalent metrics in the research community.

### 3.1.1 Notation

We consider the multi-class classification problem of predicting an output variable $\mathbf{Y}$ given an input variable $\mathbf{X}$ while remaining unbiased with respect to some variable $\mathbf{Z}$. We refer to $\mathbf{Z}$ as the protected variable. In this setting, the predictor $\hat{\mathbf{Y}} = f(\mathbf{X})$ can be constructed from tuples $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ generated from an underlying distribution $\mathcal{D}$.

### 3.1.2 Demographic parity

A predictor $\hat{\mathbf{Y}}$ satisfies *demographic parity* if $\hat{\mathbf{Y}}$ and $\mathbf{Z}$ are independent, i.e $P(\hat{\mathbf{Y}} = \hat{y})$ is equal for all values of the protected variable $\mathbf{Z}$:

$$P(\hat{\mathbf{Y}} = \hat{y}) = P(\hat{\mathbf{Y}} = \hat{y} | \mathbf{Z} = z) \quad \forall z \tag{1}$$

Given $\epsilon \in [0, 1]$, this equality is often relaxed in practice as:

$$\frac{P(\hat{\mathbf{Y}} = \hat{y} | \mathbf{Z} = z)}{P(\hat{\mathbf{Y}} = \hat{y} | \mathbf{Z} = z_{max})} \geq 1 - \epsilon \quad \forall z, \text{ where } z_{max} = \arg\max_{z} P(\hat{\mathbf{Y}} = \hat{y} | \mathbf{Z} = z) \tag{2}$$

The $p\%$ rule is defined as satisfying this inequality when $\epsilon = 1 - p/100$ .

This definition has an important societal impact, including in the field of technology law. For instance, the U.S. *Equal Employment Opportunity Commission* define the "80%-rule" , stating that a selection rate for a certain group less than 80% of that of the group with the highest selection rate, will generally be regarded by the Federal enforcement agencies as evidence of adverse impact [4].

Nevertheless, satisfying the *demographic parity* definition might have a considerable impact on the overall accuracy of the model to the point where the latter is not usable. Indeed, the latter disregards any possible correlation between the label $\mathbf{Y}$ and the protected attribute $\mathbf{Z}$. This becomes very problematic when the base rates, i.e $P(\mathbf{Y} = y|\mathbf{Z} = z)$, differ significantly across values of the protected variable as the perfect predictor is dismissed. Additionally, as the target attribute $\mathbf{Y}$ is not taken into account, the concept of *laziness* is tolerated, stating that the definition could still be satisfied by performing random classification on samples in a certain group.

### 3.1.3 Equalized odds

A predictor $\hat{\mathbf{Y}}$ satisfies *Equalized odds* if $\hat{\mathbf{Y}}$ and $\mathbf{Z}$ are conditionally independent given $\mathbf{Y}$. This means that, for *all possible values* of the true label $\mathbf{Y}$, $P(\hat{\mathbf{Y}} = \hat{y})$ is the same for all values of the protected variable:

$$P(\hat{\mathbf{Y}} = \hat{y}|\mathbf{Y} = y) = P(\hat{\mathbf{Y}} = \hat{y}|\mathbf{Z} = z, \mathbf{Y} = y) \quad \forall z, y \tag{3}$$

Unlike demographic parity, *Equalized odds* takes into account the target attribute $\mathbf{Y}$ and thus does not reject the perfect predictor $\hat{\mathbf{Y}} = \mathbf{Y}$. In addition, the definition has the ability to guarantee equality of treatment among sub-populations. Thereby, they also sanction laziness, which was one of the flaws of demographic parity [7].

### 3.1.4 Equality of opportunity

In many applications of binary classification [5, 6], one cares more about the true positive rate than the true negative rate. As a consequence, many tasks within the fairness community has focused on the following definition:

A predictor $\hat{\mathbf{Y}}$ satisfies *Equality of opportunity* with respect to a class $y$ if $\hat{\mathbf{Y}}$ and $\mathbf{Z}$ are independent conditioned on $\mathbf{Y} = y$. This means that, for a *particular value* of the true label $\mathbf{Y}$, $P(\hat{\mathbf{Y}} = \hat{y})$ is the same for all values of the protected variable:

$$P(\hat{\mathbf{Y}} = \hat{y}|\mathbf{Y} = y) = P(\hat{\mathbf{Y}} = \hat{y}|\mathbf{Z} = z, \mathbf{Y} = y) \quad \forall z \tag{4}$$

A major flaw common to both *Equality of opportunity* and *Equalized odds* is that neither of them takes into consideration the possible discrimination outside of the

model. As some applications have long been affected by various harmful biases, the proportion of samples in the privileged class could well be very different across protected variable $\mathbf{Z}$. A model may well satisfy one of the two definitions, it will still reproduce these proportions and will not close the gap in the long term [7].

### 3.1.5 Individual Fairness

One could argue that equalizing simple statistical parity between groups in each outcome class is intuitively unfair at the individual level. Indeed, pairs of individuals who are otherwise identical but differ in a protected characteristic could be assigned different outcomes. As this scenario is not penalized by group fairness, [8] introduced the notion of individual fairness:

Given a universe of individuals $\mathcal{U}$, a classification task $\mathcal{T}$ with outcome set $\mathcal{O}$, a decision model $f : \mathcal{U} \to \Delta(\mathcal{O})$ and two distance metrics: $d : \mathcal{U} \times \mathcal{U} \to \mathcal{R}$ and $D : \Delta(\mathcal{O}) \times \Delta(\mathcal{O}) \to \mathcal{R}$ as well as two thresholds $\epsilon \geq 0$ and $\delta \geq 0$, the model is *individually fair* if and only if:

$$d(u, v) \leq \epsilon \longrightarrow D(f(u), f(v)) \leq \delta, \quad \forall u, v \in \mathcal{U} \tag{5}$$

The intuition behind individual fairness is that small or non-significant perturbations of a sample ($u$, $v$ s.t. $d(u, v) \leq \epsilon$) must not be treated differently by a fair model. The choice of the input distance function $d$ identifies the perturbations to be considered non-significant, while the choice of the output distance function $D$ limits the changes allowed to the perturbed output in a fair model [10]. The difficulty of this approach lies in the choice of an appropriate distance metric as it is often hard to quantify the difference between individuals. To some extent, the problem of establishing fairness is reduced to the problem of establishing a fair distance function, which is equally challenging in certain settings.

### 3.1.6 Disparate Treatment vs Disparate Impact

Anti-discrimination laws in many countries prohibit unfair treatment of people based on sensitive attributes [18]. These laws typically evaluate the fairness of a decision making process by means of two distinct notions: *disparate treatment* and *disparate impact*.

Disparate treatment, often referred to as intentional discrimination, occurs when the model's decisions are (partly) based on the subjects' sensitive attribute information. Disparate impact, often referred to as unintentional discrimination, occurs when the

outcomes of a model that appears to be neutral in the training process disproportionately hurt (or benefit) people with certain sensitive attributes.

One should seek to design a model with design decision making systems free of disparate treatment as well as disparate impact. However, controlling for both forms of unfairness simultaneously is challenging. One could avoid disparate treatment by ensuring that the decision making process does not have access to sensitive attribute information (and its proxy attributes). Nevertheless, ignoring the sensitive attribute information may still lead to disparate impact in outcomes. As it is often the case that a model is trained on historical data, if a group with a certain sensitive attribute value was unfairly treated in the past, this unfairness may persist in future predictions through indirect discrimination, leading to disparate impact. Similarly, avoiding disparate impact in outcomes by using sensitive attribute information while making decisions constitutes disparate treatment, as the protected information is used during the model's training procedure [4].

## 3.2 Bias detection

Given a large dataset that involves human decision-making, how could one figure out if equal treatment is achieved across all demographic groups ? Previous work has attempted to deal with that issue, but the solutions found so far have mostly been task specific. One common approach is through the use of clustering, where each cluster would gather individuals that lie in the same demographic group. In that case, fairness is achieved by ensuring similar treatment across clusters. The difficulty of that approach lies in having a clustering that correctly partitions the data. Indeed, one must represent data with embeddings that correctly synthesize the relevant information to perform the task, i.e. the bias exposure of the samples. The following sections describe some of the previous techniques that have addressed this issue.

### 3.2.1 Representation Learning

When preforming the task of learning data representations, it is essential to have some control regarding the information you wish your representations to embed. [33] proposes an unsupervised learning approach to extract useful representations from high-dimensional data, which they call *Contrastive Predictive Coding*. They use an autoregressive model and a probabilistic contrastive loss which induces the latent space to capture information that is maximally useful to predict future samples. In their paper, they test the representations that result from their approach on the following tasks: speech, images, text and reinforcement learning in 3D environments.

In [34], the authors propose a framework to train de-biased representations by encouraging its outputs to be different from a set of representations that are biased by design. They assume that each image has a signal $S$ which is enough to predict the label $Y$, and a bias signal $B$ which should not influence the label $Y$. In this scenario, they capture bias B by defining a set of models $G$ that are biased by design. For example, to capture texture bias, $G$ would be the set of CNNs with small receptive fields (the outcomes are likely to be based on texture). Thereafter, they extract the biased representations from the last layer of the models. Once the representations are constructed in that way, clustering should identify the common sources of bias in the dataset. The main issue with that approach is that capturing bias through model architecture is not always possible, especially when seeking for human biases.

Similarly, [35] defines an approach that aims at detecting and quantifying social bias learned automatically from unlabeled images. It does so by implementing an image embedding association Test ($iEAT$), and calculating the distance from an image em-

bedding to certain stimuli images representing concepts. The way they choose these stimuli images is by adhering as closely as possible to stimuli defined and employed by well-validated psychological studies. As a result, they are able to assign a score to *category-concept* associations (e.g. Gender-Career, Gender-Science etc ...) and quantify the fairness of data representations. In their work, they evaluate representations from unsupervised image models pre-trained on ImageNet, iGPT and SimCLRv2. After having defined a set of stimuli images defining concepts, one could find image representations where each component represents the iEAT score to a specific concept. Clustering on these representations should then partition the dataset with the intended purpose. The main issue with this approach is the reliability of these scores, as the majority of the information on the images will not be relevant for the task (i.e. background, brightness etc ..).

### 3.2.2 Clustering

There are two distinct approaches to fair clustering; *i)* enforcing that each cluster is equally balanced in terms of sub-populations; *ii)* having each cluster represent a subpouplation of the dataset and imposing equal treatment on each. In this section, we are going to look at previous work on each approach.

[36] studies the question of fair clustering, where the sub-populations must have approximately equal representation in every cluster. They reduce the problem of fair clustering to that of classical clustering via a preprocessing step that ensures that any resulting solution is fair. They introduce the concept of *fairlets*, which are minimal sets that satisfy fair representations while approximately preserving the clustering objective and show that any fair clustering problem can be decomposed into first finding good fairlets, and then using existing machinery for traditional clustering algorithms. [38] further generalizes this work by allowing users to input the desired fair parameters (e.g. the maximum over-representation, the minimum under-representation of any group in any cluster, etc ..). Furthermore, they allow individuals to lie in multiple protected groups so that there is no need for the protected groups to partition the data.

In order to measure the fairness performance, previous work evaluates disparity in model performance between protected groups and advantaged groups in the test corpus. [39] argues that evaluating bias at the corpus level is not enough to understand how biases are embedded in a model. In fact, it could be the case that similar performances are achieved across different groups of the dataset while the model behaves differently on instances in local regions. They propose a model (i.e. $LOGAN$) based on clustering to detect these *local biases*. This is achieved by adding a term to the

$k$-means objective. Not only should the clusters minimize the distance between samples and centroids, but also maximize the difference in accuracy within each cluster between protected and non-protected groups. As a result, this enables us to explain the accuracy difference within a cluster. In the case of this research, the sensitive attributes of the dataset will not be given, and thus the identification of bias by adding a penalty term to the objective in not possible.

Clustering can also be used to not only detect bias in a dataset but also to mitigate it. [41] proposes a simple algorithm for diagnosing and mitigating bias in a dataset. In short, the algorithm consists of clustering a dataset, training a classifier on a subset of the latter and testing its performance on the clusters of the remaining data points. If the classifier performs poorly on a specific cluster relative to the others, they identify that cluster as the location of the bias. They then mitigate it by adding additional data points to the training set which are members of the biased cluster. They test their algorithm on multiple datasets, including the *Adult Census Dataset* (Section 5.1) and the *MNIST* dataset. They mention that even though their clustering procedure is very simplistic, they are able to find some explainable characteristics in the biased clusters. In particular, they discovered that 83.7% of the data points in the biased cluster of the Adult dataset identified as black males. These findings suggest that applying clustering directly on the raw dataset is enough to detect prejudices but further investigation (Section 6) shows that this is just a particular case. In reality, the representations on which the clustering is performed need to embed some information about the level of bias of the samples in order to get a useful partitioning of the data.

## 3.3 Bias mitigation

Fairness-aware machine learning algorithms seek to provide methods under which the predicted outcome of a classifier operating on individuals' data is non-discriminatory on specific protected attributes. Recent efforts in the field have produced a range of innovative techniques to improve fairness, which can be organized into the widely accepted framework of preprocessing, in-processing, and post-processing methods. In this section, we discuss the recent advancements of each.

### 3.3.1 Preprocessing

This technique seeks to modify the training data as to remove information which might result in unfair decisions, while trying to keep modifications minimal. The motivation is the idea that input data is the cause of the discrimination that a machine learning algorithm might learn, thus modifying it will prevent unfairness. In many cases, models are trained on historical datasets, some of which have shown biases against certain groups [27, 12]. Harmful bias arising from large datasets have two major causes: training data bias and overfitting bias. The former refers to instances where one can find consistent discrimination towards certain groups in the training set, while the latter refers to the unbalanced representation of these groups [27]. Once the dataset has been modified, it could be used for diverse tasks without having to alter the task-specific model itself.

Imposing fairness at the preprocessing stage consists of feeding a fair representation of the data into the algorithm. In [13], the authors find a probabilistic mapping of each individual, represented as a data point in a given input space, to a probability distribution in a new representation space. The aim of this new representation is to get rid of any information that can identify whether the person belongs to the protected subgroup, while retaining as much other information as possible. For any sample $X$ with protected attribute $S$, their mapping finds a fair representation $Z$ by maximizing the *mutual information* $I(X,Z)$ but minimizing $I(Z,S)$. Finally, they also optimize these representations so that any classification tasks that use them are maximally accurate.

The above approach has shown to satisfy demographic parity but has the downside of not being able to control the trade-off between accuracy and fairness. As previously discussed (Section 3.1), many notions of fairness are possible, and it is rarely possible to simultaneously satisfy all of them. Therefore, the ability to effectively choose between multiple notions of fairness is crucial to fair representation learning. The authors of [14] builds on [13] to provide a solution to this issue, by implementing

an information-theoretically motivated objective for learning maximally expressive representations subject to fairness constraints. Furthermore, it enables the user to control the fairness of representations by specifying limits on unfairness and providing theoretical guarantees.

On the other hand, [15] directly focuses on the training set and defines their *Classification with No Discrimination (CMD)* model, which seeks to change the dataset in a minimal way as to remove the existing discrimination. To this end, the approach first involves a model that makes predictions with no access to the protected attributes, then use a ranking function to switch the class labels of the most likely discriminated and favored samples to improve the balance of the dataset.

Instead of changing the labels, [16] suggests to assign weights to samples in the dataset. By carefully choosing the weights, the training set can be made discrimination-free w.r.t. to a specific sensitive attribute without having to change any of the labels. They consider the dataset to be discrimination-free w.r.t to a sensitive attribute once it is independent to the outcome class, i.e., the proportion of favorable label samples is equivalent across groups. As a result, their reweighting algorithm assigns weights to samples according to their group membership as to satisfy the above constraint (Section 5.2.2).

As preprocessing methods seek to alter the dataset directly, the preprocessed data can be used for any downstream tasks without having to alter the task-specific model. This is particularly useful for applications where one wants to address discrimination without having direct access to the model. In practice, this set of methods does not offer the best accuracy-fairness trade-off, which is why one needs to consider the following two approaches.

### 3.3.2 Optimization at training time

Modifications brought to specific learning algorithms, e.g. in the form of additional constraints, might be the most intuitive and has been by far the most common approach to achieve fair results. Such methods can be used to optimize for any fairness definition, hence their popularity.

The authors in [8] are among the first to formulate the question of fairness as an optimization problem. The authors develop an ambitious framework which attempts to achieve both group and individual fairness. In their setup, the goal is to define a probabilistic mapping from individuals to an intermediate representation such that the mapping achieves the two concepts. However, the main drawback of their ap-

proach is that a distance metric that defines the similarity between the individuals is assumed to be given, they do not provide intuition as to how such a metric is computed (Section 3.1.5).

In [4], the authors' goal is to design convex margin-based classifiers like logistic regression and support vector machines that avoid both disparate treatment and disparate impact (Section 3.1.6). They introduce a novel measure of decision boundary unfairness as a tractable proxy of disparate impact: the covariance between the sensitive attributes and the (signed) distance between the subjects' feature vectors and the decision boundary of the classifier. The authors claim that their measure additionally satisfies several properties: (i) for a wide variety of convex margin-based classifiers, it is convex and can be readily incorporated in their formulation without increasing their complexity; (ii) it allows for clear mechanism to trade-off fairness and accuracy; and, (iii) it can be used to ensure fairness with respect to several sensitive attributes.

A fair classifier can also be obtained through adversarial techniques, such as in [1]. In their approach, a predictor tries to predict the label $Y$ from its sample $X$, while an adversary attempts to deduce the protected attribute from the predicted label. The two models are trained simultaneously as a two-player game, such that a fair predictor results from the training process. By adjusting the information that the adversary has accessed to during training, this technique can be used to optimize for different fairness definitions, namely: Demographic parity, Equality of opportunity and Equalized odds (Section 3.1 for the definitions, Section 5.2.3 for the detailed approach).

One of the main advantages of achieving fairness by optimization at training time is the flexibility of the approach. In addition of being able to achieve good accuracy and fairness results, it allows a greater control on the accuracy-fairness trade-off (although specific to the algorithm). These algorithms are thus able to comply to anti-discrimination laws [18] while still accommodating the "business necessity". In addition, the sensitive attributes do not need to be accessed at test time, which can be of significant importance, e.g. for privacy purposes. Unlike the preprocessing approach, this technique is task specific and requires the alteration of previously designed models, which may not always be possible.

### 3.3.3 Post-processing

A third approach to building fairness into algorithm designs is by modifying the results of a previously trained classifier to achieve the fairness objectives. Such methods attempt to edit posteriors in a way that satisfies fairness constraints. The general

methodology of post-processing algorithms is to take a subset of samples and change their predicted labels appropriately to meet a group fairness requirement.

In [20], the way the authors choose the samples whose predictions are to be modified is by designing an individual bias detector, which finds samples whose model predictions change when the protected attributes change, leaving all other features constant. Once an individual bias detector is found, a post-processing bias mitigation algorithm is used to account for both individual and group fairness. Each sample from the unprivileged group is tested for individual bias against the previously defined detector. If it is likely to embed individual bias, then this sample is assigned the outcome it would have received if it were in the privileged group.

Central to the approach of [21] is the *Receiver operating characteristic* (ROC), which captures the false positive and true positive (equivalently, false negative) rates at different thresholds. These are curves in a two dimensional plane, where the horizontal axis is the false positive rate of a predictor and the vertical axis is the true positive rate. This measure is used to find an optimal threshold $R$ that yields an equalized odds predictor (all protected groups cross at the same point of the curve, i.e. the same point in the false/true positive plane) as to satisfy the *Equality of opportunity* definition (Section 3.1.4).

We say a probabilistic classifier is well-*calibrated* when the predicted probability matches the true probability of the event of interest. There are several post-processing methods for producing calibrated outputs from classification algorithms. [24] passes outputs through a learned sigmoid function, transforming them into calibrated probabilities. [25] learns a general monotonic function from outputs to probabilities. [22] is used to mitigate bias by optimizing over calibrated classifier score outputs to find probabilities with which to change output labels. Lastly, [26] seeks to identify and remove disparate impact. They describe an approach that transforms the input dataset so that predictability of the protected attribute is impossible, while preserving much of the signal of the unprotected attributes.

Post-processing methods can be applied after any classifier, without having to modify the model. This is a very useful propriety as the access to the algorithm is not always possible. However, these approaches require the sensitive attribute information at test time, which has privacy concerns. In addition, they don't allow flexibility with regards to the accuracy-fairness trade-off.

### 3.3.4 Federated Learning

Fairness is closely linked to recent advances in *Federated Machine Learning*, where an algorithm is trained across multiple decentralized edge devices holding local data samples. Indeed, non-IID data[2] widely exists in federated machine learning due to the massive distributed clients, which causes a distribution shift between training and testing data. Authors of [27] attempt to correct this shift by weighting the dataset of each client, and claim that the framework naturally leads to a notion of fairness. On the other hand, the authors of [28] believe that the problem of an unknown test distribution is better addressed by using a reweighting function applied to each training sample individually rather than on the entire dataset of clients. They apply a kernel reweighting function to each sample and train the model with the customized objective to account for any possible testing distribution. The methods described in this thesis have a similar approach to [28] but is applied in a fairness context.

We aim at developing a reweighting system able to score an image in terms of its exposure to different sources of bias. In a second stage, the weights will be used to train a fair classifier. The aforementioned approaches have the common assumption of perfect knowledge of the protected attributes. This is not the case in many computer vision tasks, as the number of subgroups increase exponentially with the sensitive attributes in consideration. For instance, in the case where gender is the protected attribute, the classifier should not only ignore direct gender features (e.g. hair, face, ...) but also indirect features (e.g. handbag, jewelry, ...) [4]. Our approach should also deal with the case where bias is static rather than explicit. For instance, many frog images are taken in swamp scenes, but a swamp itself is not a frog. Nonetheless, a model will exploit this bias if it yields mostly correct predictions. If the bias is sufficient to achieve high accuracy, there is little motivation for the the model to learn the complexity of the intended task. Our models should account for this issue and adapt when the bias shifts [19].

[2]In this context, data that is not Independent and identically distributed (IID) refers to the fact that the datasets held by independent clients are not identically distributed.

# 4 Method

Datasets can contain bias which unfairly disadvantage certain groups, and classifiers trained on these datasets can reproduce these unwanted biases. They mainly manifest themselves in two forms: i) training data bias where one can find consistent discrimination towards certain groups in the training set and ii) overfitting bias, where there is an unbalanced representation across the demographic groups of the data [27]. In this section, we seek to show that training on the reweighted dataset corresponds to training on an unobserved and unbiased data distribution that leads to a fair classifier. Our procedures can be applied to most learning algorithms and used to enforce most fairness definitions (see Section 3.1). In the hope of clarifying our explanations, let us consider as a running example the classification problem described in Section 5.1, where the aim is to predict the level of income (i.e. low, high) of an individual based on several of their attributes.

## 4.1 Cluster reweighting

In this section, we describe one of the main algorithms of this thesis. It attempts to iteratively re-weights the dataset as to enforce a specific fairness definition while maintaining a good overall accuracy. Algorithm 1 provides the pseudo-code as to satisfy *Equalizing odds* (Section 3.1.3), but we discuss the basic modifications that could be made to the procedure to enforce other fairness definitions.

Consider the classification task on a training set of size $N$ with classes $\mathcal{C}$ (e.g. the two income classes), and classification procedure $\mathcal{H}$, which can be any training procedure which minimizes a weighted loss function over some parametric function class (e.g. logistic regression, see Section 2.1). Furthermore, let $\mathcal{P}$ define the sub-populations for which the model must equalize its performance, (e.g. the two gender populations). Let $\mathbf{C} = \{C_c\}_{c=1..|\mathcal{C}|}$ define the set of clusters for each class, and $C_c = \{C_{cp}\}_{p=1..|\mathcal{P}|}$ define the partitioning into distinct sub-populations within a specific class. Note that for a given training set $\mathcal{D}$:

$$\mathcal{D} = \bigcup_{c=1}^{\mathcal{C}} \left( \bigcup_{p=1}^{\mathcal{P}} C_{cp} \right)$$

$$P(x \in C_{cm} \cap x \in C_{cn}) = 0 \qquad \forall x \in \mathcal{D}, \forall m, n \in \{1, .., |\mathcal{P}|\}, m \neq n \; \forall c \in \{1, .., |\mathcal{C}|\}$$

Intuitively, the idea is that if the accuracy of a cluster $C_{cp}$ is lower than the overall accuracy of clusters in the same class, then the corresponding cluster weight is increased as to give more importance to the correct classification of its samples at the next iteration. In our running example, if our algorithm records a lower accuracy

**Algorithm 1:** Cluster reweighting algorithm for Equalizing odds

**Input**: The partitioning $\mathbf{C}$ defined in sec.4.1, the classification procedure $\mathcal{H}$, the number of iterations $\mathcal{T}$, the learning rate $\eta$.

Initialize $\mathbf{W} = \left\{\{w_{cp}\}_{p=1..|\mathcal{P}|}\right\}_{c=1..|\mathcal{C}|}$, where $w_{cp} = 1$;

**for** $t = 1, .., \ \mathcal{T}$ **do**

  update $h$, $\mathcal{L}_h = \mathcal{H}(\ \bigcup C_{cp}, \mathbf{W}\ )$

  $\mathcal{A} = \left\{\{\ accuracy(h, C_{cp})\ \}_{p=1..|\mathcal{P}|}\right\}_{c=1..|\mathcal{C}|}$

  $\mathcal{G} = \left\{\{\ average(\{\ g_i \mid g_i \in \nabla_{\mathbf{W}}\mathcal{L}_h \ and \ x_i \in C_{cp}\})\ \}_{p=1..|\mathcal{P}|}\right\}_{c=1..|\mathcal{C}|}$

  update $\mathbf{W} = update\_weights(\ \mathcal{A}, \mathcal{G}, \eta\ )$ (See Procedure)

  update $\mathbf{W} = normalize(\mathbf{C})$

**end**

**return**: $h$

**Procedure** *update_weights*$(\ \mathcal{A}, \mathcal{G}, \eta\ )$

  **for** $w_{cp}, a_{cp}, g_{cp}$ *in* $\mathbf{W}, \mathcal{A}, \mathcal{G}$ **do**

    **if** $a_{cp} < average(A_c)$ **then**

      $w_{cp} \longleftarrow w_{cp} + \eta g_{cp}$

    **else**

      $w_{cp} \longleftarrow w_{cp} - \eta g_{cp}$

    **end**

  **end**

  **return**: $\mathbf{W}$

in the women' cluster compared to the men' cluster of the high income class, it will increase the weight of the former before its next iteration. Similarly, if the accuracy of a cluster $C_{cp}$ is higher than the overall accuracy of clusters in the same class, its corresponding cluster weight is decreased as to free up some of the weight distribution. Note that the training procedure $\mathcal{H}$ will seek to minimize the weighted loss function, and thus will tend to increase the overall model accuracy.

Algorithm 1 works by iteratively performing the following step: (1) Obtain the new classifier and weighted loss resulting from the training procedure evaluated with the cluster weights from the previous iteration; (2) Compute the accuracy of the classifier in each cluster; (3) Compute the average gradient of each cluster w.r.t the cluster weights of the loss function; (4) Update the cluster weights w.r.t the cluster accuracies and gradients (Procedure of Algorithm 1); (5) Normalize the cluster weights.

Let $\mathbf{A} = \{A_c\}_{c=1..|\mathcal{C}|}$ define the set of cluster accuracies for each class, and $A_c = \{A_{cp}\}_{p=1..|\mathcal{P}|}$ define the accuracies of each sub-population cluster within a specific class. The procedure of Algorithm 1 gives the pseudo code describing the weight update

step. The direction of the cluster weight update depends on the cluster accuracy: its weight will increase if its accuracy is under the average accuracy of the clusters in the same class, and will decrease otherwise. Unsurprisingly, the update value is the product of the (average) cluster gradient and the learning rate.

In step (5), the updated weights are normalized as to distribute the weight distribution equally across classes. If this step were to be skipped, cluster weights would eventually have different magnitudes, resulting in large differences in fairness performance across classes. For instance, if 95% of the weight distribution is distributed among the low income class, then the latter would (almost) have identical model performance across sub-populations, but drastic differences in the high income class. This case is not desirable when optimizing for *Equalizing odds*, thus the normalizing step is essential. The pseudo code of the normalization method along with the *accuracy* and *average* methods is trivial and is not provided here.

As previously mentioned, the algorithm can be easily adapted to enforce different fairness definitions. For instance, *demographic parity* (see Section 3.1.2) is an easier case. The definition requires the predictions and sensitive attribute(s) to be independent, and does not take into account the actual label. Consequently, the clustering $\mathbf{C}$ does not need to differentiate between classes; i.e. $\mathbf{C} = \{C_p\}_{p=1..|\mathcal{P}|}$. In our running example, this means that the clustering needs only to distinguish between the two genders and not the income classes. Furthermore, *Equality of opportunity* only looks at the positive samples in binary classification. Thus, the partitioning of the dataset would only take positive samples into consideration; i.e. $\mathbf{C} = \{C_{1p}\}_{p=1..|\mathcal{P}|}$ (e.g. exclusively the men and women within the high income class).

Our resulting algorithm simultaneously minimizes the weighted loss and maximizes fairness via learning the cluster weights. This may be interpreted as competing goals with different objective functions. Thus, one can view the algorithm as a form of a non-zero-sum two-player game, where the adversary tries to select the cluster weights to maximize the loss of the objective, whereas the learner tries to find the best model parameters to minimize the worst case loss over the sources of bias in the dataset [28].

**Algorithm 2:** Sample reweighting algorithm for Equalizing odds

**Input**: The partitioning $\mathbf{C}$ defined in sec.4.1, the classification procedure $\mathcal{H}$, the number of iterations $\mathcal{T}$, the learning rate $\eta$.

Initialize $\mathbf{W} = \{w_i\}_{i=1..N}$, where $w_i = 1$;

**for** $t = 1, .., \ \mathcal{T}$ **do**

  update $h$, $\mathcal{L}_h = \mathcal{H}(\ \bigcup C_{cp}, \mathbf{W}\ )$

  $\mathcal{A} = \left\{\{\ accuracy(h, C_{cp})\ \}_{p=1..|\mathcal{P}|}\right\}_{c=1..|\mathcal{C}|}$

  update $\mathbf{W} = update\_weights(\ \mathbf{C}, \mathcal{A}, \mathcal{H}, \nabla_{\mathbf{W}}\mathcal{L}_h, \eta\ )$ (See Procedure)

  update $\mathbf{W} = normalize(\mathbf{C})$

**end**

**return**: $h$

**Procedure** *update_weights*$(\ \mathbf{C}, \mathcal{A}, \mathcal{H}, \mathcal{G}, \eta\ )$

  **for** $c \in \{1, \ldots, |\mathcal{C}|\}, p \in \{1, \ldots, |\mathcal{P}|\}$ **do**

    **for** $x_i \in \mathbf{C}_{cp}$ **do**

      **if** $a_{cp} < average(A_c)$ *and* $h(x_i)$ *is not* $c$ **then**

        $w_i \longleftarrow w_i + \eta g_i$

      **end**

      **if** $a_i > average(A_c)$ *and* $h(x_i)$ *is* $c$ **then**

        $w_i \longleftarrow w_i - \eta g_i$

      **end**

    **end**

  **end**

  **return**: $\mathbf{W}$

## 4.2 Sample reweighting

The algorithm described in Section 4.1 allocates the same weight for every sample lying in the same cluster. This means that the weight of misclassified samples lying in a cluster for which the accuracy is above average will decrease. As a result, decreasing the weight of a misclassified sample will deteriorate the chance of its correct classification at the next iteration. Similarly, the weight of correctly classified samples lying in a cluster for which the accuracy is under average will increase. Increasing the weight of correctly classified samples is not needed and is a waste of the weight allocation. For the above reasons, having the same weight for the entire cluster is restraining and can lead to negative repercussions on the overall accuracy of the model.

In this section, we discuss a modified version of algorithm 1, where instead of giving a weight per cluster, we give a weight to every sample of the dataset. We give the

pseudo-code of the modified algorithm in Algorithm 2; note that the inputs remain unchanged. The main changes of the procedure are threefold: (1) The structure of $\mathbf{W}$, as it now represents sample weights instead of cluster weights; (2) The step involving computation of cluster gradients is skipped as they are no longer needed to update the weights; (3) The weight update step, for which the pseudo-code is given in the procedure of Algorithm 2. The cluster accuracies and normalization steps are identical.

Giving a weight to every sample gives more freedom to the model to accommodate fairness while keeping a good overall accuracy. This idea is illustrated in the procedure of Algorithm 2, where we describe how to update the sample weights. Intuitively, the main difference with the weight update procedure of Algorithm 1 is that we can update the weight of a sample depending on the correctness of its classification. In particular, within a cluster for which the accuracy is above the average accuracy of clusters in the same class, we only decrease the weights of those samples that are correctly classified by the current classification model $h$. In our running example, consider that the men' cluster has a higher accuracy than the women' cluster in the high income class. In this case, instead of decreasing the weight of every men within the class, we will decrease the weight of only those that are correctly classified by $h$, while the weight of the misclassified men will remain unchanged. Similarly, within a cluster for which the accuracy is under the average accuracy of clusters in the same class, we only increase the weights of those samples that are misclassified. This modification benefits the overall accuracy of the model while still enforcing the fairness constraint (See Section 5.3).

While Algorithm 2 provides more freedom, it comes at the cost of being computationally more expensive than Algorithm 1. As every sample has its own weight, the weight update and normalizing steps have to iterate through the entire dataset. As a result, the training process is slow, in particular for large datasets. In addition, our sample reweighting algorithm is not as interpretable as the cluster reweighting algorithm, as having a weight for every sample makes the comparison between clusters more cubersome. In the case of Algorithm 1, the cluster weights could be manually examined to understand to degree of prejudice in the dataset.

These two algorithms are evaluated against tabular and image datasets in Section 5 and 6.

## 4.3 Clustering images for bias detection

Although the fairness community has significantly grown recently, many questions are still to be answered. Indeed, even though the research in the field has received a lot of attention in supervised algorithms [35, 21, 5], the research is not as extensive when data is not fully disclosed, e.g. when the protected attributes are not clearly defined. This issue is emphasized in the image domain for two main reasons: (i) higher-level concepts are often inferred, for example from a collection of pixels, rather than stated explicitly as a feature and (ii) since causes of bias are not limited to sensitive attributes, brute force attempts to generate finely labeled data tend to be infeasible in terms of efficiency or even replicate existing bias [49]. In this case, imposing some notion of fairness first requires the model to recognize and cluster together data with similar attributes, and identify the presence of protected attributes in the image. In this section, we will consider the binary classification task described in Section 6, which consists of classifying the sport represented in the images, namely *basketball* and *volleyball*.

This section is motivated by the insight that there is a need for a framework to impose fairness when the protected attributes are not explicitly stated. To our knowledge, existing work in the field assumes complete labeling of protected attributes. This work aims to bring a novel solution in identifying sources of bias in an image dataset for which the groups that ought to be protected are not explicit, as well as imposing group fairness constraints to achieve fair classification.

We define signal $S$ and bias $B$ as cues for the recognition of input image $X$ as a certain target variable $Y$. Signal $S$ is the essential cue for the recognition of $X$ as $Y$; examples include the shape and texture of the ball for sport classification. Bias $B$, on the other hand, are cues that are not essential for the recognition but can be correlated with the target $Y$; e.g. the jersey color is not essential for sport recognition and can be regarded as bias $B$ (see Section 6). A key property of $B$ is that modifying this signal does not change $Y$; changing the jersey color of a player does not change the sport they are playing.

We wish to capture bias $B$ by finding image representations that encapsulate the signal. Although not explicitly given, we can approximate $B$ by defining of model $\mathcal{M}$ that is biased towards $B$ by design. For texture bias, for example, we define $\mathcal{M}$ to be a Convolutional Neural Network (CNN) architecture with small receptive fields. Then, the classifications will be made based on the patterns that this structure manages to detect (i.e. textures), becoming more liable to overfit to texture [34]. In other cases, constraining the complexity of $\mathcal{M}$ and training it for a very small number of epochs

is enough to restrain its ability to generalize to unseen data and overfit to the bias, as $B$ often represents simpler cues for solving the task.

Upon completion of the training process, the embeddings are extracted from the last layer of $\mathcal{M}$. As these are the embeddings located right before classification happens, they must put in evidence the most prominent cues that distinguish between classes, i.e. the bias signal $B$. At this point, clustering could be applied to the extracted representations. Note that, in order for the clusters to partition the data in terms of $B$, the embeddings must not only encapsulate as much of $B$ as possible, but also disregard the valid signal $S$. Hence, we choose to further enrich this process with an intermediate step; dimension reduction via *Principal Component Analysis* (see Section 2.3). The main motivation of this additional step is to retain exclusively the main sources of variance in the dataset and ignoring noisy signals (e.g. image background, image lighting, ...) as well as obtain disentangled representations. This intermediate step has shown to significantly improve the clustering.

The last step of the procedure is to perform clustering on the transformed data. We apply the *k-means* clustering algorithm (section 2.4) on each class to partition the data, with the customed distance metric:

$$d = | < \lambda, \mathbf{v_1} - \mathbf{v_2} > | \qquad \forall\ \mathbf{v_1}, \mathbf{v_2} \in \mathbb{R}^n \tag{6}$$

where $< \cdot >$ and $| \cdot |$ are the scalar product operator and the absolute value respectively and $\lambda$ is the eigenvalue vector resulting from the PCA reduction. The magnitude of the eigenvalue $\lambda_i$ is directly linked to the proportion of the total variability that principal component $i$ explains. Hence, it would not be fair that these proportions are not taken into account when clustering; by incorporating the eigenvalue vector to the *k-means*' distance metric, we ensure that the images that are similar w.r.t the principal decomposition axes are clustered together. As a concrete example, consider PCA dimensionality reduction on sport classification data. Suppose that the first dimension represents the player's jersey color and explains 95% of the data variance. Applying *k-means* with our customized distance metric will ensure that same colored jersey images are clustered together and that the rest of the dimensions do not disproportionately impact the clusters.

**Algorithm 3:** Clustering procedure

**Input**: Data $\mathbf{X}$ defined in sec. 4.3, the model $\mathcal{M}$, the number of iterations $\mathcal{T}$, the number of dimensions $d$, the number of clusters per class $k$.

update $\mathcal{M} = train(\ \mathcal{M}, \mathbf{X}, \mathcal{T}\ )$

**for** $X_c$ *in* $\mathbf{X}$ **do**

$\tilde{X}_c \longleftarrow extract(\ \mathcal{M}, X_c\ )$

$Z_c, \lambda_c \longleftarrow PCA(\ \tilde{X}_c, d\ )$

$K_c \longleftarrow K\_mean(\ Z_c, \lambda_c, k\ )$

**end**

**return**: $\mathbf{K} = \{k_c\}_{c=1,\dots,|\mathcal{C}|}$

Consider data $\mathbf{X} = \{X_c\}_{c=1,\dots,|\mathcal{C}|}$, where $\mathcal{C}$ represents the classes of the classification task. Algorithm 3 summarizes the clustering procedure. The *train* method refers to the training procedure of model $\mathcal{M}$ and the *extract* method refers to the embedding extraction from the last layer of $\mathcal{M}$. One could wonder the reason for using a different PCA reduction for each class, as this adds additional complexity to the procedure. In the general case where one reduction is used for the entire dataset $\mathbf{X}$, the variance that the reduction seeks to explain includes the differences between the classes. As the clustering algorithm is applied to every class sequentially, this behavior is undesirable in our setting, i.e. we prefer that the PCA reductions focus on explaining the variance within every class separately. Lastly, note that the *k_means* algorithm is applied with the customized distance metric (equation 6) mentioned previously.

# 5 Experiments with Tabular data

## 5.1 Adult Data Set

This dataset is credited to Ronny Kohavi and Barry Becker [31] and was drawn from the 1994 United States Census Bureau data, it involves using personal details to predict whether an individual's yearly income exceeds $50,000 per year. The features used for predictions include the age, the gender, the education level and the race (see a complete list of the features as well as an explanation of the preprocessing used in Appendix 8.1). The dataset contains 48'842 samples of which 3'620 contain missing values. As our approach is not purely driven by the model's final accuracy, we choose to ignore samples containing missing values and are left with 45'222 samples to use for training and testing.

As previously mentioned, the data is divided into two classes; the individuals earning more $50,000 (positive class) and those who are earning less (negative class), which makes this task a binary classification problem. The classes are imbalanced and only contains about 25% of positive samples. We choose to down sample the negative class as to convert the task to a balanced classification problem. The reason is that we do not want the imbalances of the dataset to be an additional factor explaining the potential unfairness in the data.

In this section, we are going to set *gender* as the sensitive attribute and will extend this set with the *race* attribute when looking at subgroup fairness in Section 5.4. The dataset is imbalanced in terms of the gender attribute; the men to women ratio being 62.4% - 37.6% in the negative class and 85.1% - 14.9% in the positive class. From these numbers, it is clear that one of the biases is the under-representation of women (especially in the positive class), which will need to be protected in order to get similar classification accuracy for every subgroup. The bias is even more present with respect to the binary *race* attribute (white vs non-white), where the ratio is 83.9% - 16.1% in the negative class and 91.1% - 8.9% in the positive class.

We split the dataset into a training and test set. We choose to intentionally change the sample distribution of the test set as to make it balanced with regards to the sensitive attribute(s). Testing the model on a balanced set will put forward the bias learnt from the severely imbalanced training data.

## 5.2 Comparative models

In this section, we enumerate existing models that we use for comparison to our algorithms. Apart from the *Vanilla* model, these algorithms seek to eradicate the differences between the enumerated sub-populations of the dataset to achieve fair classification. They each lie into a different fairness-seeking framework described in Section 3.3 and are briefly described here.

### 5.2.1 Vanilla

As a baseline, we trained a logistic regression model (Section 2.1) on the previously described dataset, without any further constraints. This shows us the performance of a simple model in terms of overall accuracy and provides us with a lower bound on fairness, i.e. the treatment discrepancies when the training process is left unconstrained.

### 5.2.2 Reweighting preprocessing

This discrimination-aware preprocessing method was first proposed by [16] and consists of *reweighting* the data to remove discrimination without relabeling instances. They stipulate that by carefully choosing the weights at preprocessing time, the training dataset can be made discrimination-free w.r.t. the sensitive attribute without having to change any of the labels.

Consider the binary classification problem on dataset $\mathbf{D}$ with a binary sensitive attribute $S$ with domain $\{b,\ w\}$ and target attribute *Class* with domain $\{-, +\}$. "+" is the desirable class for the data subjects and the objects satisfying $S = b$ and $S = w$ represent, respectively, the deprived and the favored community.

If the dataset $\mathbf{D}$ is unbiased, i.e., $S$ and *Class* are statistically independent, the expected probability $P_{exp}(S = b \cap Class = +)$ would be:

$$P_{exp}(S = b \cap Class = +) = \frac{|X \in \mathbf{D}|X(S) = b|}{|\mathbf{D}|} \times \frac{|X \in \mathbf{D}|X(Class) = +|}{|\mathbf{D}|}$$

In reality, however, the observed probability in D,

$$P_{obs}(S = b \cap Class = +) = \frac{|X \in \mathbf{D}|X(S) = b \cap Class(X) = +|}{|\mathbf{D}|}$$

which may be different. If the observed probability is lower than the expected probability, this means that dataset $\mathbf{D}$ contains bias towards the samples where $X(S) = b$.

To correct for possible bias, each sample is assigned the weight:

$$W(X) = \frac{P_{exp}(S = X(S) \cap Class = X(Class))}{P_{obs}(S = X(S) \cap Class = X(Class))}$$

In this way, a weight is assigned to every tuple according to its $S$ and *Class* values. It is easy to see that the transformed dataset is unbiased; i.e. if we multiply the frequency of every object by its weight, we measure no difference between the sensitive attributes. The fair classifier is learnt on this discrimination-free dataset.

Although our approach also re-weights the dataset, the significant difference is that they are dynamically adapted through the training process. Reweighting the dataset at preprocessing time only corrects for unbalances between sub-populations, which is not the only form in which bias can manifest itself. The code of this model is taken from [17] and had to be re-adapted to fit our dataset.

### 5.2.3 Adversarial debiasing in-processing

This bias mitigation technique was first proposed by [1] and seeks to achieve fairness through its training process. They present a framework for mitigating biases simultaneously learning a predictor and an adversary. The input to the predictor $\mathbf{X}$ produces a prediction $\mathbf{Y}$ while the adversary tries to model a protected variable $\mathbf{Z}$. Then, the objective is to maximize the predictor's ability to predict $\mathbf{Y}$ while minimizing the adversary's ability to predict $\mathbf{Z}$.

More formally, they define a *predictor*, trained to accomplish the task of predicting $\mathbf{Y}$ given $\mathbf{X}$ by modifying weights $\mathbf{W}$ to minimize some loss $L_p(\hat{y}, y)$ using a gradient-based method such as stochastic gradient descent. Furthermore, the output layer of the predictor is then used as an input to another network called the *adversary* which attempts to predict sensitive attribute $\mathbf{Z}$. The adversary has loss term $L_A(\hat{z}, z)$ and weights $\mathbf{U}$. Although this approach can be used to satisfy many fairness definitions, we will focus on the *Equalized odds* definition for comparison with our algorithms. As a result, the adversary needs as input the prediction $\hat{\mathbf{Y}}$ and true label $\mathbf{Y}$.

During the training process, the adversary's weights $\mathbf{U}$ are updated at each iteration according to gradient $\nabla_U L_A$. Furthermore, the predictor's weights $\mathbf{W}$ are updated according to the following expression:

$$\nabla_W L_P - proj_{\nabla_W L_A} \nabla_W L_P - \alpha \nabla_W L_A$$

The first term of the expression, $\nabla_W L_P$ is necessary as to reduce the predictor's loss $L_p$. The second term $proj_{\nabla_W L_A} \nabla_W L_P$ is present as to prevent the weight update

from moving in a direction that will end up helping the adversary. The last term of the expression $\nabla_W L_A$ is used to make sure that the updates hurt the adversary. $\alpha$ is a tuneable hyper-parameter.

Note that the training procedure is simliar to the adversarial training of a *Generative Adversarial Network* [43], where the adversary can be interpreted as the *discriminator*. Such two-player optimization procedures have shown to be unstable in the past, and the reproducibility of the results is definitely a major drawback of this approach. Nevertheless, we use the code given in [2] to apply this framework to our dataset and compare the results with our algorithms in Section 5.3.

### 5.2.4 Receiver Operating Characteristic post-processing

[22] is a post-processing method that investigates the relationship between calibration and error rates. We say that a model is *calibrated* if within the set of people who receive a predicted probability of $p$, we have a $p$ fraction of the members of this set that are positive instances of the classification problem. For many applications such as risk assessment in the criminal justice, this property is key as in the oppositive case, having a probability estimate of $p$ could carry a different meaning across demographic groups, which amounts to having different treatment. They provide a simple post-processing algorithm that withholds predictive information for randomly chosen inputs to achieve parity and preserve calibration.

Let $P \subset \mathbb{R}^k \times \{0, 1\}$ be the input space of a binary classification task. Additionally, we assume the presence of two groups: $G1, G2 \subset P$, which represent disjoint population subsets, such as different races. We assume that the groups have different *base rates* $\mu_t$ (section 5.5), defined as the probabilities of belonging to the positive class: $\mu_1 = P_{(x,y)\sim G1}[y = 1] \neq P_{(x,y)\sim G2}[y = 1] = \mu_2$. Furthermore, let $h_1, h_2 : \mathbb{R}^k \to [0, 1]$ be binary classifiers, where $h_1$ classifies samples from $G1$ and $h_2$ classifies samples from $G2$. Each classifier outputs the probability that a given sample $x$ belongs to the positive class. Finally, we let $c_p(h_t)$ and $c_n(h_t)$ be the false positive and false negative rate of classifier $h_t$. They define a *Relaxed Equalized Odds with Calibration* definition that their algorithm is trying to enforce.

Given a cost function $g_t = a_t c_p(h_t) + b_t c_n(h_t)$, classifiers $h_1$ and $h_2$ achieve *Relaxed Equalized Odds with Calibration* for groups $G_1$ and $G_2$ if both classifiers are calibrated and satisfy the constraint $g_1(h_1) = g_2(h_2)$. It is worth noting that the *Equalizing odds* definition constraints $c_p(h_t)$ and $c_p(h_t)$ to be equal across groups.

**Algorithm 4:** Achieving Calibration and an Equal-Cost Constraint via Information Withholding

**Input** Classifiers $h_1$ and $h_2$ s.t. $g_2(h_2) \leq g_1(h_1) \leq g_2(h^{\mu_2})$, holdout set $P_{valid}$.

- Determine base rate $\mu_2$ of $G_2$ (using $P_{valid}$) to produce trivial classifier $h^{\mu_2}$.
- Construct $\tilde{h}_2$ using $\alpha = \frac{g_1(h_1) - g_2(h_2)}{g_2(h^{\mu_2}) - g_2(h_2)}$, where $\alpha$ is the interpolation parameter.

**Return**: $h_1$, $\tilde{h}_2$ — which are calibrated and satisfy $g_1(h_1) = g_2(\tilde{h}_2)$.

Given $h_1$ and $h_2$ with $g_1(h_1) \geq g_2(h_2)$, they seek to arrive at a calibrated classifier $\tilde{h}_2$ for group $G_2$ such that $g_1(h_1) = g_2(\tilde{h}_2)$. They show that this cost constraint can be achieved by withholding predictive information for a randomly chosen subset of group $G_2$.

$$\tilde{h}_2(x) = \begin{cases} h^{\mu_2} = \mu_2 & \text{with probability } \alpha \\ h_2(x) & \text{with probability } 1 - \alpha \end{cases}$$

Notice that the cost of the new classifier $\tilde{h}_2$ is now a linear combination of the two others; i.e. $g_2(\tilde{h}_2) = \alpha g_2(h^{\mu_2}) + (1 - \alpha) g_2(h_2)$, meaning that the *Relaxed Equalized Odds with Calibration* condition $g_1(h_1) = g_2(\tilde{h}_2)$ is achieved with:

$$\alpha = \frac{g_1(h_1) - g_2(h_2)}{g_2(h^{\mu_2}) - g_2(h_2)}$$

Their procedure is summarized in Algorithm 4.

There are two main limiting factors to this approach. The first one is that the procedure equalizes cost by making a classifier worse for one of the groups. Second, it achieves this cost increase by withholding information on a randomly chosen population subset, making the outcome inequitable within the group. This raises the issue of *individual fairness* (section 3.1.5), as the (random) misclassifications added by Algorithm 4 will always be tragic to the individuals who are affected.

| | | | Low Income | | High Income | | |
|---|---|---|---|---|---|---|---|
| | Training Acc. | Test Acc | Males | Females | Males | Females | Abs. Difference |
| Vanilla (5.2.1) | 0.799 | 0.784 | 0.644 | 0.906 | 0.877 | 0.709 | 0.2150 |
| Pre-Proc. Reweighting (5.2.2) | 0.797 | 0.809 | 0.674 | 0.800 | 0.879 | 0.881 | 0.0640 |
| Adv. Debiasing (5.2.3) | 0.806 | 0.827 | 0.807 | 0.878 | 0.784 | 0.841 | 0.0640 |
| Post-Proc. ROC (5.2.4) | 0.824 | 0.822 | 0.829 | 0.924 | 0.765 | 0.769 | 0.0495 |
| Cluster Reweighting (4.1) | 0.797 | 0.806 | 0.734 | 0.805 | 0.807 | 0.876 | 0.0705 |
| Sample Reweighting (4.2) | 0.807 | 0.820 | 0.771 | 0.858 | 0.810 | 0.841 | 0.0590 |

Table 1: Experimental results of the comparative and reweighting models on the Adult dataset, which is split class-wise in terms of the sensitive attribute *gender*. The first two rows represent the training and test accuracy. The next four rows represent the test accuracy for each group of each class. The last column represents the average of the absolute differences between clusters of each class.

## 5.3 Experiments

In this section, we evaluate the aforementioned models on the *Adult Dataset* (see Section 5.1). Table 1 demonstrates the experimental results. As previously mentioned, we choose the *gender* attribute as the protected attribute. The table's first two rows represent the training and test accuracy. The next two rows split individuals belonging to the negative class (i.e. low income class) according to their gender and records the test accuracy. Similarly, the next two rows do the same for individuals belonging to the positive class (i.e. high income class). Note that although the training set is unbalanced, the test set was designed so that the aformentioned four groups contain the same number of samples (we forward the reader to Section 5.1 for a deeper description of the data distribution). Lastly, the last row represents the average of the absolute differences between the two groups of each class. Each experiment is ran for 160 epochs and the underlying model of each is Logistic Regression (see Section 2.1).

The first model we evaluate is the Vanilla model, which gives lower bounds on the fairness performance that should be achieved by future models. First, we notice that the model does not generalize on a balanced test set as its accuracy is lower than the training one. Furthermore, we see a significant difference between the two groups of each class; i.e. the women are significantly more likely to be classified in the negative class, while the men are more likely to be classified in the positive class. The significant average difference of 21.5% expresses the unfairness of the model.

The next three models described in Section 5.2 demonstrate significant improvements on fairness, while keeping both training and test accuracy high. Each of them has a higher test than training accuracy, meaning that they generalize well to unbiased

and unseen data. Both *Pre-proc Reweighting* and *Adversarial Debiasing* achieve an average difference of 6.4% while *Post-Proc. ROC* achieves a difference of 4.95%.

The two last rows of the table correspond to the algorithms of Section 4. In this setting, the number of classes $\mathcal{C} = 2$ and number of sub-populations $\mathcal{P} = 2$ yield a total of 4 clusters. Furthermore, $\mathcal{T} = 160$ and the classification procedure $\mathcal{H}$ is the training procedure of Logistic Regression (see Algorithms 1 & 2). These models both generalize well to out of sample data, the *Cluster Reweighting* model has an absolute difference of 7.05% while the *Sample Reweighting* model has a difference of 5.9%. As explained previously, the freedom gained from adding a weight per sample explains the increase in test accuracy from the former to the latter model. Furthermore, we also observe that these added weights help increasing fairness between groups in the test set.

While the three comparative models perform relatively well in these settings, they each present specific drawbacks which make their use difficult in practice (see Section 5.2). Furthermore, one common limitation to these algorithms is that they do not extent to subgroup fairness. Indeed, they do not support having more than one sensitive attribute. In the case of algorithms described in Section 4, having one sensitive attribute is only a specific case of the approach. The data can have an arbitrary number of classes and be clustered to any granularity by changing the structure of the clustering parameter $\mathbf{C}$. We demonstrate a case of subgroup fairness in the next section.

| | | | Low Income | | | | High Income | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Training Acc. | Test Acc | FD | FW | MD | MW | FD | FW | MD | MW | Max. Diff |
| Vanilla (5.2.1) | 0.801 | 0.760 | 0.929 | 0.913 | 0.775 | 0.623 | 0.503 | 0.688 | 0.742 | 0.871 | 0.337 |
| Cluster Reweighting (4.1) | 0.798 | 0.801 | 0.773 | 0.797 | 0.759 | 0.731 | 0.880 | 0.868 | 0.812 | 0.780 | 0.083 |
| Sample Reweighting (4.2) | 0.807 | 0.807 | 0.836 | 0.830 | 0.892 | 0.702 | 0.841 | 0.850 | 0.804 | 0.785 | 0.123 |

Table 2: Experimental results of the reweighting models on the Adult dataset, being split class-wise in terms of the sensitive attributes *gender* and *race*. The first two rows represent the training and test accuracy. The next eight rows represent the test accuracy per cluster for each of the two classes. The last column represents the average of the maximal differences between clusters of each class.

## 5.4 Towards Subgroup Fairness

Assume we wish to use the previous group fairness analysis (i.e. single protected attribute) to evaluate our model's fairness w.r.t both *gender* (male vs female) and *race* (white vs non-white) separately, both of which are assumed to be distributed independently and uniformly at random in our dataset. Consider a classifier that assigns the positive class if and only if the individual is a non-white female or a white male. Evaluating fairness w.r.t the two previous binary attributes separately will lead us to conclude that the model is perfectly equitable, in the sense that it labels both male and female as well as white and non-white as positive 50% of the time. However, if one looks at any conjunction of the two attributes (e.g. white female), it is evident that the classifier maximally violates any statistical parity fairness constraints. Consequently, subgroup fairness is an important notion that must be considered when evaluating a model.

Table 2 illustrates the experimental results on the Adult dataset. The first two rows represent the training and test accuracy. The next eight rows represent the test accuracy per cluster for each class, with the following column codes; FD - non-white females; FW - white females; MD - non-white males; MW - white males. As in the previous section, the test set is designed so that these eight clusters are of the same size. The last column represents the average of the maximal differences between clusters of each class.

We first evaluate the Vanilla model, which illustrates the bias that will be reproduced by models trained on the training data if no precautions are taken. Again, we notice that the Vanilla model does not generalize on a balanced test set. Furthermore, we observe a gradual performance increase across the columns in the positive class. This illustrates bias, with the most discriminated group being non-white females. In

addition, the last column tells us that this bias is significant, as the average maximal difference in accuracy between groups of each class is about 33.7%.

The next two rows are defined in the following setting. While the number of classes is still $\mathcal{C} = 2$ , the number of sub-populations increases to $\mathcal{P} = 4$, which yields a total of 8 clusters. Furthermore, $\mathcal{T} = 160$ and the classification procedure $\mathcal{H}$ is still the training procedure of Logistic Regression. While these models both generalize well to out of sample data (i.e. Test acc $>$ Training acc), we observe that the test accuracy tends to be lower than in the group fairness case. This tendency is explained by the increase of sub-populations; constraining the model to equalize performances across a larger number of subgroups will come at the cost of a decrease in performance of the highest performing clusters, along with a decrease in overall performance. However, we note that while the number of clusters has doubled, the performance decrease of our algorithms is only 0.5% and 1.3% respectively. As of previously, the test accuracy of the *Sample reweighting* algorithm is still higher than the one of the *Cluster reweighting* algorithm.

Subgroup fairness interpolates between group and individual notions of fairness. The semantics of statistical notions of group fairness are significantly stronger when defined over a large number of subgroups, thus allowing a middle ground between fairness for a small number of pre-defined groups, and the strong assumptions needed for fairness at the individual level (each sample having its own cluster).

## 5.5 Base rate Analysis

One may consider that the simplest approach to designing a fair algorithm is to exclude protected characteristics from the data. In our case, the choice of using the protected attributes (*gender* and *race*) as part as the individuals' embeddings was well thought about. Moreover, we demonstrate by an example that discriminatory behavior is sometimes reinforced by the omission of sensible information.

Consider the example of pretrial recidivism (Compas, [46]), where one must assign a score to the criminal defendant's likelihood of reoffending. Throughout history, the evidence shows that women reoffend less than men in many jurisdictions [47]. Consequently, insisting upon a model that is blind with regards to the criminal's gender will consistently overstate the recidivism risk of women, which introduces significant bias. By acknowledging the predictive value of gender in this setting, one could create a decision rule that detains fewer people (particularly women) while achieving the same public safety benefits. Conversely, by ignoring this information and basing decisions solely on the gender-neutral risk assessments, one would be engaging in taste-based discrimination against women [44].

When the sensitive attributes add predictive value, excluding this information will in many cases lead to unjustified disparate impact. When this is not the case, one could decide to exclude all protected attributes for predictions. However, we note that in the case that sensitive information has no influence on the outcome, one could theoretically include exclusively samples from a single protected group (white men) in the training set. With enough samples, the model should generalize across all demographic groups and having a balanced training set should not influence the fairness of the predictions. This argument presents an inconsistency in many informal discussions of fairness, as advocates often argue for both a balanced dataset as well as exclusion of sensitive information at training time.

In the case of a gender-wise split Adult dataset, we hypothesise that perfectly equal statistics across groups is perhaps not desirable. The motivation is that if one group is subsequently harder to classify that the other, the model would have to purposely hurt samples from the easier group to obtain fairness, which is not the goal. Consider again the example of pretrial recidivism, where the base rate of a certain group represents the group's recidivism rate. [44] demonstrates the effect of increasing the base rate of a sub-population on the false positive rate. They argue that higher false positive rate (unjustified detention rate) for one group relative to another may either mean that the former group faces stricter standards of detention, indicating discrimination, or, alternatively, that the group has a higher base rate. In other words, increasing

| | Base models | | | Sample Re. model | | |
|---|---|---|---|---|---|---|
| | Class 0 | Class 1 | Average | Class 0 | Class 1 | Average |
| Original | 0.047 | 0.042 | **0.0445** | 0.012 | 0.107 | **0.0595** |
| D5% | 0.132 | 0.142 | **0.137** | 0.072 | 0.187 | **0.1295** |
| D10% | 0.173 | 0.192 | **0.1825** | 0.086 | 0.207 | **0.1465** |
| D15% | 0.212 | 0.205 | **0.2085** | 0.179 | 0.208 | **0.1935** |

Table 3: Experimental results highlighting the effect of altering the base rate difference on our *Sample Reweighting* algorithm on the Adult dataset. The leftmost column indicates the dataset used, where Dx% expresses that $x$% of the easiest/hardest classifiable samples were removed from the original dataset. Each cell of the Class 0 and Class 1 columns represents the difference of classification accuracy between the groups (males vs females) in the indicated class.

the base rate difference between two sub-populations must have an impact on the model' statistical parity measures, or else that would indicate evidence for discriminatory behavior towards the group with the higher base rate.

On that note, we evaluate our *Sample Reweighting* model when altering the base rate difference as illustrated in Table 3. The leftmost column indicates the dataset used, where Dx% expresses the degree of modification from the original dataset. It is modified by training a toy model on the original dataset and examining its classification confidence for each sample. In order to increase the base rate difference between males and females, the male samples classified with largest confidence are removed from the training set (as to decrease the male's base model performance) along with the female samples classified with lowest confidence (as to increase the female's base model performance). As a result, the (average) base difference between the two sub-population increases along with the percentage of removed samples. In the case of our algorithm, we note that the (average) difference between the sub-populations also increases at approximately the same rate. As previously mentioned, this indicates that our model does not blindly enforce equal treatment on the two sub-populations and partially takes into account the base rate of each separately.

A common critique of group fairness definitions is that they are satisfiable by retaining the discriminatory behavior but randomly switching some classification outcomes as to meet the constraints. Previously described post-processing ROC model (see Section 5.2.4) is a good example where this conduct would not be sanctioned. Of course, this radically violates fairness at the individual level and will by no means

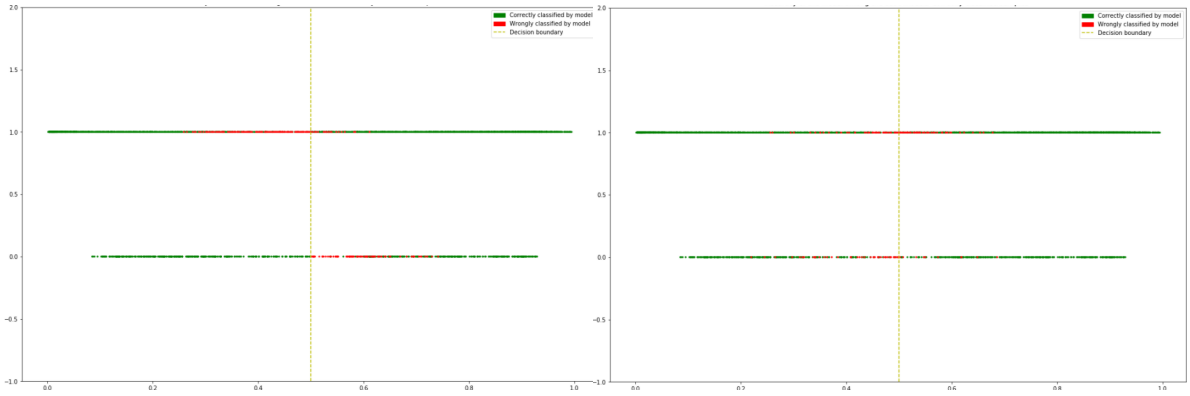



Figure 3: Analysis of misclassified samples of Vanilla (left) and Sample reweighting (right) models. The left plot underlines bias in the model as the red zones are not centered around the classification threshold. The right plot shows correction of this bias as the red zones are re-centered.

eradicate unfair behavior. Next, we investigate whether in the pursue of discrimination-free predictions, our algorithm falls into the trap of violating some of the fundamentals of individual fairness.

Figure 3 seeks to clarify the above doubts. Each dot represents a sample that has been correctly classified by its corresponding base model; its x-value gives the confidence of the classification and its y-value indicates the sub-population it belongs to (0 - female, 1 - male). The dot color refers to the classification correctness of the model that is being evaluated (*Vanilla* model on the left figure, *Sample Reweighting* model on the right figure); green and red colors stand for correct and incorrect classification respectively. The yellow dotted line indicates the classification threshold; each sample lying on the left of the line belongs to the negative class while samples lying on the right belongs to the positive class. As an example, a red sample lying on the upper line and right side of the yellow threshold, means that (1) the sample corresponds to a male individual (upper line); (2) it has been correctly classified in the positive class by the male's base model ($x_i > 0.5$); (3) it has been misclassified in the negative class by the model that is being evaluated (red color).

The left plot of figure 3 corresponds to the *Vanilla* model. We first observe that the red dots are concentrated around their mean and do not spread to the tails of the distributions. This illustrates that samples which are misclassified by the *Vanilla* model have base model logits which do not allow clear classification (around 0.5). Within the male sub-population, the red zone leaning towards the left of the threshold highlights evidence of disproportionate false positive rate; male individuals

that belong to the negative class ($x_i < 0.5$) are misclassified (red color) by the *Vanilla* model into the positive class. Conversely, the red zone of the female sub-population leaning towards the right of the threshold highlights evidence of disproportionate false negative rate; female individuals that belong to the positive class ($x_i > 0.5$) are misclassified (red color) by the *Vanilla* model into the negative class. These asymmetries confirm the presence of bias in the model.

The right plot of figure 3 corresponds to the *Sample Reweighting* model. The first difference with the leftmost plot is the reduction of the asymmetries of the red zones; the zones have re-centered around the threshold. This provides evidence that the two sub-populations are treated in a similar way due to the corrections made by the algorithm. We further observe that the red zones are still concentrated around their mean, having only a few outliers further from the zone. This observation clarifies our original doubt; even though the *Equalizing Odds* metric has decreased from 21.5% to 5.9% (see Table 1), the samples that are misclassified by our model are also challenging to the corresponding base models and are clearly not randomly picked for the purpose of satisfying the fairness definition.

# 6 Experiments with Image data

## 6.1 Sport Dataset

The sport dataset is an image dataset composed of 1643 samples, split into two classes, namely a *basketball* and *volleyball* class. The classes are balanced (i.e. number of samples in the two classes are similar) and we divide the dataset into a training and test set (75% - 25% split). The images consist of in-game pictures, portraits and team pictures in their respective discipline (sample examples can be found in Appendix 8.2.1). Furthermore, we give details about the preprocessing of the images in Appendix 8.2.2.

As previously mentioned, the images are divided in two (balanced) classes, making this task a binary classification problem. For the sake of exploring fairness in our approach, we manually divided the classes according to the *jersey color* and *gender* of the player(s) represented in the images, which we use as sensitive attributes for the rest of the section. More specifically, we manually label each picture as being part of the *red* or *yellow* jersey color and *male* or *female* gender group, which enables us to later evaluate our algorithm in terms of this attributes. Note that these groups have originally the same number of samples in both classes (i.e. basketball and volleyball). In order to introduce bias in the dataset, we modify the training set as to contain four times as many red (resp. male) samples as yellow (resp. female) in the basketball class, as well as four times as many yellow (resp. female) samples as red (resp. male) in the volleyball class; i.e. we introduce an *80%* bias as the ratio, *red:yellow* (resp. *male:female*), becomes *4:1* and *1:4* in the basketball and volleyball class respectively. We introduce this bias with the aim of fooling the model into classifying the samples based on insignificant features (e.g. jersey color, gender) instead of sport specific attributes (i.e. ball texture, pitch, ...). Note that these modifications are applied to the training set only, while the test set remains with balanced proportions.

In Section 6.2, we evaluate the performance of our algorithms on this modified dataset. In Section 6.3, we adjust the level of bias in the dataset to further evaluate our algorithms' performances. In section 6.4, we develop a semi-supervised approach to better partition the dataset in terms of the two sensitive attributes.

| | Clustering (%) | | | | | | | | Algorithm | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Basket | | | | Volley | | | | | | | | | |
| | Cluster 1 | | Cluster 2 | | Cluster 1 | | Cluster 2 | | | Basket | | Volley | | |
| | R | Y | R | Y | R | Y | R | Y | Test Acc | R | Y | R | Y | Abs. Diff. |
| Vanilla | · | · | · | · | · | · | · | · | 0.819 | 0.937 | 0.707 | 0.670 | 0.958 | 0.258 |
| Pre. Rewei. (5.2.2) | · | · | · | · | · | · | · | · | 0.851 | 0.93 | 0.786 | 0.811 | 0.81 | 0.075 |
| Cluster Rewei. (4.1) | 100 | 0 | 0 | 100 | 100 | 0 | 0 | 100 | 0.853 | 0.848 | 0.869 | 0.806 | 0.895 | 0.054 |
| Sample Rewei. (4.2) | 100 | 0 | 0 | 100 | 100 | 0 | 0 | 100 | 0.831 | 0.786 | 0.778 | 0.835 | 0.937 | 0.054 |
| Cluster Rewei. (4.1) | 63.3 | 36.7 | 100 | 0 | 4.1 | 95.9 | 93.7 | 6.3 | 0.846 | 0.839 | 0.818 | 0.806 | 0.926 | 0.069 |
| Sample Rewei. (4.2) | 63.3 | 36.7 | 100 | 0 | 4.1 | 95.9 | 93.7 | 6.3 | 0.846 | 0.866 | 0.848 | 0.757 | 0.916 | 0.086 |

Table 4: Experimental results showing the performance of the Vanilla, preprocessing reweighting and our models on the Sport dataset. The second section of the table assumes perfect clustering in terms of sensitive attribute *jersey color* red (R) and yellow (Y). In the third section of the table, we use procedure 4.3 to find the clusters (see 6.2.1) and evaluate the performance of our algorithms.

## 6.2 Experiments

In this section, we evaluate our algorithms on the Sport dataset. Table 4 demonstrates the performance of multiple models, divided into three sections each containing two models. The first eight columns of the array represent the clustering, where each class (basketball vs volleyball) is partitioned into two clusters. The models in the second section of the table assume perfect clustering, i.e. the two clusters perfectly partition each class in terms of red (R) and yellow (Y) jersey colors. The last section uses procedure 4.3 to find a relevant clustering (see Section 6.2.1 for implementation details). The last six columns are highlighting the accuracy and fairness of different algorithms with their respective clustering (see Section 6.2.2).

Note that unlike tabular data (section 5) where the underlying model is logistic regression, we choose to use a pre-trained *Resnet* in this section (see Section 2.2 for a description of the model).

### 6.2.1 Clustering

In the first section of Table 4 (Vanilla and preprocessing reweighting models), the concept of clustering is not relevant to their procedure. Indeed, the Vanilla model only includes regular training without taking the sensitive attribute into account. Additionally, the preprocessing reweighting model only needs access to the samples' protected attributes and labels at preprocessing time but then applies regular training on the reweighted data. However, clustering is crucial in the procedure of the cluster

and sample reweighting models (input $\mathcal{C}$ in procedures 4.1 and 4.2) as it dictates the subgroups that the models will seek to equalize in their respective algorithm.

As mentioned previously, the second section of the table evaluates the aforementioned models assuming perfect clustering of the dataset. Evidently, this procedure is impossible without access to the sensitive attribute of each sample. In the third section of the table, the clustering is performed according to the procedure described in Section 4.3. In this setting, we wish to find embeddings which capture the jersey color bias. We apply Algorithm 3 in order to find a relevant clustering. We define input model $\mathcal{M}$ to be a small CNN (see Appendix 8.3.2 for model definition), the number of iterations $\mathcal{T} = 5$, the number of reduction dimensions $d = 10$ and the number of clusters per class $k = 2$. Note that while this procedure modifies the training procedure of the algorithms, it does not alter the test data and evaluation methods in any way.

From the third Section of Table 4, one can see that Algorithm 3 manages to capture the bias of the dataset in its resulting clusters. Indeed, the first cluster of the basketball class includes all yellow colored images, while the second contains exclusively red colored samples. The contrast is even larger in the volleyball clusters, as the first one contains an overwhelming majority of yellow colored samples while the second almost exclusively red ones. Unlike the second section of the table, this clustering is done without any knowledge of the sensitive attribute, which puts forward a tremendous benefit for *Computer Vision* tasks.

### 6.2.2 Model evaluation

Table 4 also shows the performance of each model in terms of accuracy and fairness. The first model we evaluate is the Vanilla model, which achieves a test accuracy of 81.9%. When looking at the fairness results, one can observe the obvious difference of treatment in favor of red basketball and yellow volleyball samples. This disparity is due to the severe bias voluntarily established in the dataset. Furthermore, we can note from the last column that the average difference in performance is significant (i.e 25.8%). The preprocessing reweighting model described in Section 5.2.2 significantly improves test accuracy as well as fairness and gives us good baseline results for future comparisons with our models.

The last four rows of Table 4 correspond to our algorithms described in Section 4. In this setting, the number of classes $\mathcal{C} = 2$ and the number of sub-populations $\mathcal{P} = 2$ yield a total of 4 clusters. Furthermore, the number of iterations $\mathcal{T} = 60$ and the classification procedure $\mathcal{H}$ is the training procedure of the Residual Neural Network (see Section 2.2). When assuming perfect clustering (Section 2 of the Table 4), both

| | Clustering (%) | | | | | | | | Algorithm | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Basket | | | | Volley | | | | | | | | | |
| | Cluster 1 | | Cluster 2 | | Cluster 1 | | Cluster 2 | | | Basket | | Volley | | |
| Bias | R | Y | R | Y | R | Y | R | Y | Test Acc | R | Y | R | Y | Abs. Diff. |
| 80% | 63.3 | 36.7 | 100 | 0 | 4.1 | 95.9 | 93.7 | 6.3 | 0.846 | 0.839 | 0.818 | 0.806 | 0.926 | 0.069 |
| 70% | 51.7 | 48.3 | 97.9 | 2.1 | 9.6 | 90.4 | 87.7 | 12.3 | 0.850 | 0.804 | 0.808 | 0.825 | 0.968 | 0.071 |
| 60% | 63.2 | 36.8 | 55.8 | 44.2 | 46.6 | 53.4 | 26.3 | 73.7 | 0.875 | 0.866 | 0.919 | 0.816 | 0.905 | 0.070 |
| 50% | 52 | 48 | 58.6 | 41.4 | 50.5 | 49.5 | 46.9 | 53.1 | 0.892 | 0.812 | 0.909 | 0.913 | 0.947 | 0.067 |

Table 5: The table shows the clustering and the resulting *cluster reweighting* algorithm performance that arise as a result of changing the level of bias in the dataset. Note that $p$% bias stands for the representation proportion of the majority group within each class of the training set, i.e. red for the basketball class, yellow for the volleyball class. The first eight columns depict the clustering that results from procedure 4.3, while the last six record the accuracy and fairness performance of the cluster reweighting algorithm of Section 4.1.

accuracy and fairness significantly improve from the Vanilla model. Interestingly enough, the performance of these two algorithms only degrade by a few percents when incorporating procedure 4.3 to find the clustering. Indeed, even though the clusters now contain some noise, the two algorithms still achieve attractive fairness results. Note that these findings are very encouraging, as it highlights the fact that transitioning from having labeled to unlabeled sensitive attributes only worsens the subgroup disparities by a negligible margin.

## 6.3 Adjusting the Bias

To further investigate our algorithms, we purposely alter the level of bias in the dataset. As both Algorithm 1 and Algorithm 2 have the same mechanics, we choose to evaluate the former one in this section. Table 5 highlights the clustering that results from procedure 4.3 and the resulting performance from our cluster reweighting algorithm.

The leftmost column indicates the level of bias in the dataset; $p$% bias stands for the representation proportion of the majority group within each class of the training set, i.e. red for the basketball class, yellow for the volleyball class. Unsurprisingly, one can see that the clustering that results from the 50% and 60% bias datasets are not significant; the deviations between the clusters are attributed to noise rather than structure w.r.t the jersey color. On the other hand, the clustering that results from

the 70% bias dataset has a distinct structure; the clusters are formed in terms of the jersey color attribute. The structure is even stronger when considering the 80% bias dataset. As a result, we stipulate the following interesting point: when the bias becomes increasingly present in the dataset (i.e. $\geq 70\%$), the jersey color becomes the main source of distinction between the sample embeddings.

When reviewing the performance of the cluster reweighting algorithm, one can notice from Table 5 that the test accuracy decreases as the level of bias in the dataset increases. Indeed, increasing the bias means more constraints on the algorithm to maintain similar accuracies between the sub-populastions. In this case, maintaining a good fairness performance comes at the cost of decreasing the overall accuracy of the model. In addition, one can see that the fairness performance is very similar regardless of the strength of the bias in the dataset. This allows us to make the following important observation: no matter the level of bias in the dataset, clustering the data according to procedure 4.3 before applying Algorithm 1 will train an efficient classifier while accounting for possible bias in the data. More specifically, assume the dataset contains very little bias. Applying the cluster reweighting algorithm will yield a classifier with strong test accuracy. Furthermore, even though the clustering will not necessarily have a specific structure, the classifier will still have strong fairness results due to our starting assumption. On the contrary, assume the dataset contains significant bias. In this case, the clustering will detect the subgroup disparities and will allow Algorithm 1 to account for the unbalances in the data and yield a fair classifier.

| | | Clustering (%) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Basket | | | | Volley | | | | |
| | | Cluster 1 | | Cluster 2 | | Cluster 1 | | Cluster 2 | | |
| | | R/M | Y/F | R/M | Y/F | R/M | Y/F | R/M | Y/F | Abs. Diff. |
| **Jersey** | Perfect Clus. | 100 | 0 | 0 | 100 | 100 | 0 | 0 | 100 | 0.054 |
| | Semi-sup. Clu. (6.4) | 99.7 | 0.3 | 8.8 | 91.2 | 83.5 | 16.5 | 1.8 | 98.2 | 0.064 |
| | Unsup. Clu. (4.3) | 63.3 | 36.7 | 100 | 0 | 4.1 | 95.9 | 93.7 | 6.3 | 0.069 |
| **Gender** | Perfect Clus. | 100 | 0 | 0 | 100 | 100 | 0 | 0 | 100 | 0.096 |
| | Semi-sup. Clu. (6.4) | 89.4 | 10.6 | 55.2 | 44.8 | 41.1 | 58.9 | 6.0 | 94.0 | 0.10 |
| | Unsup. Clu. (4.3) | 81.7 | 18.3 | 68.2 | 31.8 | 18.2 | 81.8 | 23.9 | 76.1 | 0.12 |

Table 6: The table compares different clustering approaches: i) perfect clustering ii) semi-supervised clustering iii) unsupervised clustering, and demonstrates the corresponding fairness result of each approach. The last column represents the average of the absolute differences between the two categories of the evaluated protected attribute, i.e Red (R) and Yellow (Y) for **Jersey**, Male (M) and Female (F) for **Gender**.

## 6.4 Semi-supervised Clustering

As a result of the positive outcome of previous experiments, one can conclude that our unsupervised clustering approach (section 4.3) is enough to provide a relevant clustering regardless of the protected attribute. Unfortunately, we note that our procedure heavily depends on the visual features of the protected attribute. As illustrated on the sample images in Appendix 8.2.1, the *jersey color* attribute is visually very dominant (i.e takes up an important portion of the pixels of the image), resulting in a successful clustering procedure. In addition, we observe that when using a more subtle sensitive attribute, e.g *gender*, our fully unsupervised approach is not as fruitful (see Table 6). In this section, we address this problem by developing a *semi-supervised* approach that has the aim of improving the overall fairness of our cluster reweighting algorithm (section 4.1).

Manual labeling of a small sample set is achievable in almost every task. We select a small portion of the training set while making sure it is balanced w.r.t both the class and the sensitive attribute, which we manually label. On this set, we train a classifier to predict the sensitive attribute of each sample. Subsequently, we use the classifier to predict the entire training set and thus provide a relevant clustering.

In Table 6, we select 115 samples ($\sim$ 14% of the training set) and manually label *jersey color* and *gender*. Once again, we use the *Resnet* classifier (section 2.2) and use the sensitive attribute as labels. We use our classifier to predict the entire training set and present the clustering results in the table. Lastly, we use the clusters provided by the approach to run our cluster reweighting algorithm (section 4.1).

For comparison, Table 6 also shows the cluster reweighting algorithm's results when assuming perfect clustering as well as unsupervised clustering. The last column represents the average of the absolute differences between the two categories of the evaluated protected attribute, i.e Red (R) and Yellow (Y) for **Jersey**, Male (M) and Female (F) for **Gender**. As mentioned in section 6.1, we remind the reader that we introduce an *80%* bias in the training set, i.e. the ratio (*red:yellow* or *male:female*) becomes *4:1* and *1:4* in the basketball and volleyball class respectively. As expected, we observe that for both the **Jersey** and **Gender** case, the quality of the clusters increases along with the available information regarding the samples' sensitive attribute, i.e. perfect clustering assumes full disclosure, semi-supervised assumes some disclosure and unsupervised assumes no disclosure. In addition, the fairness metric resulting from applying the cluster reweighting algorithm follows the same logical trend.

The results presented in Table 6 allow us to draw the following interesting conclusions. First, we note that algorithm fairness increases as we increase the sensitive information we provide along with the data. Furthermore, this experiment reinforces the credibility of our cluster reweighting algorithm as it confirms that its performance is directly related to the quality of the clustering we feed it. Overall, we observe that the fairness performance for **Gender** is poorer than **Jersey**. This adds to the evidence that the overall fairness pipeline defined in this thesis will work best for visually dominant features as opposed to subtle ones.

# 7 Conclusion

In this thesis, we developed tools to overcome biases that arise from unbalanced or unfair datasets. Furthermore, we constructed a methodology to expose bias among unlabeled samples. In this section, we summarize our contributions, draw conclusions, and suggest directions for future work.

## 7.1 Contributions and directions for future work

In Section 4, we presented a framework for correcting dataset bias based on reweighting the training data. In a first step, we partition the data into distinct subgroups and apply our procedure to find the cluster weights that eradicate group disparities at test time while retaining a good overall accuracy. Subsequently, we reduce the clusters to training samples and demonstrate how this new framework benefits from individual reweighting. Furthermore, we discuss the flexibility of these approaches as well as the way they can be leveraged to enforce most prevalent fairness definitions. In a second step, we examine the case of *unsupervised fairness*, where we aspire to expose bias when the sample attributes are not clearly defined. To this end, we derive a methodology involving representation learning and dimension reduction to yield an advantageous clustering that allows bias mitigation at a later stage.

In Section 5, we observe that our algorithms are able to find a good trade-off between fairness and accuracy when applied to tabular datasets. We introduce common existing approaches to produce a fair classifier and show that our methods often provide either better or comparative predictive error than other fair classification methods. Furthermore, we expose the significant disadvantage of many existing fairness-seeking methods: they cannot be extended to subgroup fairness. We explain how this can be achieved with our procedures and demonstrate significant performances when extending our scheme to two protected attributes. Lastly, we ask ourselves whether we fall into the common fairness pitfall of satisfying group fairness definitions by randomly switching some classification outcomes as to meet the constraints. By using an argument involving the base rates of the dataset's sub-populations, we are able to demonstrate that this is not case in our procedures.

In Section 6, we repeat our analysis with an Image Dataset. We first assume knowledge of the samples' protected attributes and show that our models are able to find a good trade-off between fairness and accuracy. Subsequently, using our bias detection methodology to infer the samples' sensitive attributes allows us to demonstrate that not only does it expose the appropriate bias, but also that our models are robust to noisy clustering. We continue this section by analysing our detection procedure when

adjusting the level of bias in our dataset. Upon observing the results, we are able to draw a meaningful conclusion: no matter the level of bias in the dataset, using our procedure to detect plausible bias and applying one of our mitigation algorithms will train an efficient classifier while accounting for possible bias in the data. Finally, we conclude this section by evaluating our fairness pipeline with *gender* as a more subtle sensitive attribute. We observe that our unsupervised approach has more difficulties in situations where the sensitive attribute is not visually significant and develop a *semi-supervised* approach to improve the clustering in such circumstances.

Our approaches can be enhanced a number of ways. Firstly, despite providing experimental evidence, we do not provide theoretical guarantees regarding the fairness and convergence performances of our approaches. [40] evaluates a similar approach in a theoretical way. Our procedures should be investigated in a similar manner. Furthermore, our Image analysis would benefit from using a richer Image dataset. Although the Sport Dataset contains complex and universal samples, we note that it is rather small. As a possible extension, we suggest to repeat the analysis on the *UTKFace* dataset (see description in Appendix 8.4) as we believe it would provide a good insight into the mechanics of our approaches.

## 7.2 Final remarks

Let us now end by circling back to the objectives we have set ourselves in the introduction. There are many ways in which discrimination can manifest itself in society. Although the development of fair algorithms certainly helps in mitigating such injustice, we emphasize that machine learning alone cannot solve these issues single-handedly. As long as research in the field is still moving at such rapid pace, algorithms should only support or inform human decision makers and should not have the final say as to avoid undesirable effects in various social issues.

What does one mean by *undesirable effects*? Arguably, whether it be for human decision makers or researchers designing fairness-seeking algorithms, the concept of *fairness* still depends on subjective human judgment both on the level of individuals as well as societies and cultures. Ultimately, this notion is almost impossible to quantify, as is an ultimate discrimination-free algorithm.

# References


[1] Brian Hu Zhang, Blake Lemoine, Margaret Mitchell. *Mitigating Unwanted Biases with Adversarial Learning*, Stanford University, Google Mountain View, CA (2018)

[2] Samuel Hoffman, Guillermo Martín, *AI Fairness 360 (AIF360)* DOI: https://github.com/Trusted-AI/AIF360/blob/master/aif360/algorithms/inprocessing/adversarial_debiasing.py (2016)

[3] Pranay K. Lohia, Karthikeyan Natesan Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R. Varshney andRuchir Puri, *Bias mitigation post-processing for individual and group fairness*, IBM Research and IBM Watson AIPlatform (2018)

[4] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, Krishna P. Gummadi, *Fairness Constraints: Mechanisms for Fair Classification*, Max Planck Institute for Software Systems (MPI-SWS), Germany (2015)

[5] Moritz Hardt, Eric Price, Nathan Srebro, *Equality of Opportunity in Supervised Learning*, University of California, Berkeley (2016)

[6] Candice Schumann, Jeffrey S. Foster, Nicholas Mattei, John P. Dickerson, *We Need Fairness and Explainability in Algorithmic Hiring*, University of Maryland (2020)

[7] Alexandre Landeau, *Measuring Fairness in Machine Learning Models*, https://medium.com/data-from-the-trenches/measuring-fairness-in-machine-learning-models-2be070fab712

[8] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel *Fairness through awareness*. In Proceedings of the 3rd innovations in theoretical computer science conference. ACM, 214–226 (2012)

[9] Ildiko Suveg, George Vosselman, *3D reconstruction of building models*, Technical University of Delft, The Netherlands (2000)

[10] Philips George John, Deepak Vijaykeerthy, Diptikalyan Saha *Verifying Individual Fairness in Machine Learning Models* IBM Research AI (2020)

[11] Mehryar Mohri, Gary Sivek, Ananda Theertha Suresh,*Agnostic Federated Learning*, Google Research, New York(2019)

[12] Pranay K. Lohia, Karthikeyan N. Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R. Varshney and Ruchir Puri, *Bias mitigation post-processing for individual and group fairness*, IBM Research and IBM Watson AI Platform (2018)

[13] [7] Zemel, Wu, Swersky, Pitassi, Dwork. *Learning Fair Representations*, University of Toronto, Microsoft Research (2013)

[14] Jiaming Song, Pratyusha Kalluri, Aditya Grover, Shengjia Zhao, Stefano Ermon, *Learning Controllable Fair Rep-resentations*, Standford University (2018)

[15] Faisal Kamiran, Toon Calders. *Classifying without Discriminating*, Eindhoven University of Technology. (2009)

[16] F. Kamiran and T. Calders, *Data Preprocessing Techniques for Classification without Discrimination*, Knowledge and Information Systems, (2012.)

[17] Samuel Hoffman, James Budarz, *AI Fairness 360 (AIF360)* DOI: https://github.com/Trusted-AI/AIF360/blob/master/aif360/algorithms/preprocessing/reweighting.py (2020)

[18] C. Munoz, M. Smith, and D. Patil. Big Data: *A Report on Algorithmic Systems, Opportunity, and Civil Rights.* Executive Office of the President. The White House, 2016.

[19] Nikolas Adaloglou, *Intuitive Explanation of Skip Connections in Deep Learning*, AI summer, https://theaisummer.com/skip-connections/

[20] Pranay K. Lohia, Karthikeyan Natesan Ramamurthy, Manish Bhide, Diptikalyan Saha, Kush R. Varshney and Ruchir Puri, *Bias mitigation post-processing for individual and group fairness*, IBM Research and IBM Watson AI Platform (2018)

[21] Moritz Hardt Eric Price Nathan Srebro *Equality of Opportunity in Supervised Learning*, University of California, Berkeley, (2016)

[22] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, andK. Q. Weinberger, *On Fairness and Calibration* Conference on Neural Information Processing Systems, 2017

[23] Niki Kilbertus, *Beyond traditional assumptions in fair machine learning*, Department of Engineering University of Cambridge (2021)

[24] J. Platt. *Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods.* Advances in Large Margin Classifiers, 10(3):61–74, 1999.

[25] B. Zadrozny and C. Elkan. *Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In ICML, pages 609–616, 2001.*

[26] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, *Certifying and removing disparate impact.* ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015.

[27] Mehryar Mohri, Gary Sivek, Ananda Theertha Suresh, *Agnostic Federated Learning*, Google Research, New York (2019)

[28] Wei Du, Depeng Xu, Xintao Wu, Hanghang Tong. *Fairness-aware Agnostic Federated Learning*, University of Arkansas, University of Illinois, (2020)

[29] Abdalla M. El-Habib. *Multinomial Logistic Regression Model*, Faculty of Economics and Administrative Sciences, Al-Azhar University, (2012)

[30] Ron Kohavi, *Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid*, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, (1996)

[31] Ronny Kohavi and Barry Becker, Data Mining and Visualization Silicon Graphics, Donors of the *Adult dataset*

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, *Deep Residual Learning for Image Recognition*, Microsoft Research (2015)

[33] Representation Learning with Contrastive Predictive Coding, A. Oord, Y. Li, O. Vinyals, Google DeepMind, 22 Jan 2019

[34] Learning De-biased Representations with Biased Representations, H. Bahng, S. Chun, S. Yun, J. Choo, S. Oh, Korea University, 30 June 2020.

[35] Image Representations Learned With Unsupervised Pre-Training Contain Human-like Biases, Ryan Steed, Aylin Caliskan, Canergie Mellon University, USA, 2021

[36] Fair Clustering Through Fairlets, F. Chierichetti, R. Kumar, S. Lattanzi, S. Vassilvitskii, Good research, 15 Feb 2018 (NeurIPS)

[37] Nicolas Kayser-Bril.Google apologizes after its Vision AI produced racist resultshttps://algorithmwatch.org/,April 2020.

[38] Fair Algorithms for Clustering, S. Bera, D. Chakrabarty, N. Flores, M. Negahbani, Dartmouth, 17 Jun 2019

[39] LOGAN: Local Group Bias Detection by Clustering, J. Zhao, K. Chang, California Nov 2020

[40] Heinrich Jiang, Ofir Nachum *Identifying and Correcting Label Bias in Machine Learning*, Google Research, Mountain View, CA (2019)

[41] Predicting Bias in Machine Learned Classifiers Using Clustering, R. Thomson, E. Alhajjarm J. Irwin, T. Russull, Military Academy, May 2018.

[42] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer London, 2011.

[43] Goodfellow, I.; Pouget-Abadie, J.;Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. *Generative adversarial nets. In Advances in neural information processing systems*

[44] Sam Corbett-Davies, Sharad Goel. *The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning*, Standford University (2018) (2014)

[45] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, Luc Van Gool, *SCAN: Learning to Classify Images without Labels*, KU Leuven ETH Zurich (2020)

[46] Matias Barenstein, *ProPublica's COMPAS Data Revisited*, Federal Trade Commission (FTC), (2019)

[47] DeMichele, M., Baumgartner, P., Wenger, M., Barrick, K., Comfort, M., and Misra, S. *The public safety assessment: A re-validation and assessment of predictive utility and diferential prediction by race and gender in kentucky.* (2018)

[48] Angwin, J., Larson, J., Mattu, S., and Kirchner, L. Machine bias: There is software used across the country to predict future criminals. and it is biased against blacks. ProPublica, May, 23, 2016. URL https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

[49] Kyriakou, K., Barlas, P., Kleanthous, S., and Otterbacher,J. *Fairness in proprietary image tagging algorithms: Across-platform audit on people images.* In Proceedings of the International AAAI Conference on Web and Social Media, volume 13, pp. 313–322, 2019

[50] Zhang, Zhifei, Song, Yang, and Qi, Hairong, *Age Progression/Regression by Conditional Adversarial Autoencoder*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2017)

# 8 Appendix

## 8.1 Appendix A: the Adult dataset

Here, we give a description of the attributes of the Adult dataset:

- *Income*: Binary variable (>50K, ≤50K) regarded as the label.
- *Age*: Continuous variable.
- *Workclass*: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- *Education*: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
- *Education-num*: Discrete variable representing the number of years of education.
- *Marital-status*: Categorical variable with the following categories: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
- *Occupation*: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
- *Relationship*: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried. race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
- *Sex*: Female, Male.
- *Capital-gain*: continuous variable representing the capital gain of an individual.
- *Capital-loss*: continuous variable representing the capital loss of an individual.
- *Hours-per-week*: continuous variable representing the number of working hours per week.
- *Native-country*: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad & Tobago, Peru, Hong, Holand-Netherlands.

As mentioned in Section 5.1, the samples with missing values are dropped. The '>50K' class of the *income* attribute is given the positive label and '≤50K' class the negative label. The *age* attribute is divided into categories, namely: 1-18, 18-25, 25-30, 30-35, 35-40, 40-45, 45-50, 50-55, 55-60, 65-90. The categorical variables are given *one hot encondings*, namely: the workclass, occupation, education, relationship, race, martial-status and native-country. All continuous variables are scaled and the train-test ratio used in Section 5 is 0.75:0.25.

## 8.2 Appendix B: the Sport dataset

### 8.2.1 Sample examples


(a) In-game basketball image (b) Basketball player portrait c) Basketball team picture


(c) In-game volleyball image (d) Volleyball player portrait e) Volleyball team picture

Figure 4: Sample examples from the Sport dataset

### 8.2.2 Sample preprocessing

After reading the images from the dataset, they are preprocessed in order to standardize their format as to make the model's training process more efficient. The images are first resized so that they all have the same resolution level. Next, they are cropped from the center in order to standardize the range as to maintain consistency over different images. Lastly, each image is normalized so that each input feature (i.e. pixel) has a similar distribution. This makes convergence faster when training the network. The training and test set are preprocessed the same way.

## 8.3 Appendix C: Experimental settings

### 8.3.1 Tabular dataset

As mentioned in Section 5, the underlying model $\mathcal{H}$ is a logistic regression model. We choose the learning rate to be 0.001 as well as to use pytorch's adam optimizer. Regarding the weight update scheme, we choose the weight update learning rate to be 10 as well as to initialize the (cluster or sample) weights as 1. Note that all of these parameters along with many more can be inputted by the user (see Appendix 8.5 for a description of the main code packages).

### 8.3.2 Image dataset

The pytorch model $\mathcal{M}$ has the following structure:

- $Conv2D(in\_filters = 3, out\_filters = 1, kernel\_size = (4,4), stride = (1,1))$
- $MaxPool2d(kernel\_size = 2, stride = 2, padding = 0, dilation = 1, ceil\_mode = False)$
- $Linear(in\_features = 12100, out\_features = 512, bias = True)$
- $Linear(in\_features = 512, out\_features = 2, bias = True))$

In Section 6.2.1, the 512 dimension embeddings are extracted from the last layer before being reduced to 10 dimension embeddings by PCA.

As mentioned in Section 6, the underlying model $\mathcal{H}$ is a Residual Neural Network (Section 2.2). We choose the learning rate to be of exponential decay, starting from 0.002 and reducing by a factor of 0.2 every 5 epochs. We choose to use pytorch's $SGD$ optimizer. We define the (cluster or sample) weight update learning rate to be 1. Note that all of these parameters along with many more can be inputted by the user (see Appendix 8.5 for a description of the main code packages).

## 8.4 Appendix D: UTKFace dataset

UTKFace dataset is a large-scale face dataset with long age span (range from 0 to 116 years old). The dataset consists of over 20,000 face images which are aligned and cropped. The images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc... Because the images are labelled by age, gender, and ethnicity, this opens the door to many fairness tasks such as the one described in this thesis.

## 8.5 Appendix E: Description of code packages

The following document briefly discusses the main scripts of the code, as well as give the commands to reproduce the results. The full code is available at :
https://github.com/ghayat2/Agnostic_Fairness

# Towards Detection and Mitigation of Dataset Bias through Adaptive Reweighting

This project focuses on detecting and mitigating bias in tabular and image datasets. The libraries that are required to run the code include: matplotlib, pytorch, numpy, pandas, visdom (only for tracking training progress) and scikit-learn. These can be installed through pip3:

```
pip3 install <library_name>
```

This document will serve as a guide to understand the most important scripts.

## Authors

- Gabriel Hayat

## Datasets

In the `Datasets` folder, two datasets are relevant for this project:

- Adult dataset: This dataset is tabular. It involves using personal details to predict whether an individual's yearly income exceeds $50,000 per year. The features used for predictions include the age, the gender, the education level, the race etc ... The dataset contains 48'842 samples. When handling this dataset, we are going to set *gender* as the sensitive attribute and will extend this set with the *race* attribute when looking at subgroup fairness.
- basket_volley: This sport dataset is an image dataset composed of 1643 samples, split into two classes, namely a *basketball* and *volleyball* class. The classes are balanced (i.e. number of samples in the two classes are similar). The images consist of in-game pictures, portraits and team pictures in their respective discipline. For the sake of exploring fairness, we manually divided the classes according to the jersey color of the player(s) represented in the images, which we use as a sensitive attribute when handling this dataset.

Details about how these datasets are preprocessed and prepared for training can be found in the report.

## Clustering

This folder refers to the bias detection section of the project. The goal is to detect whether the dataset is biased against certain sub-population(s).

- **unsupervised_clustering.ipynb** : This notebook aims at clustering the dataset to expose potential bias. It uses a small convolutional network to find embeddings of every sample of the dataset, before applying PCA and using the k-means algorithm with a customized distance metric to yield a clustering of the dataset. This notebook can be run sequencially, cell by cell. (see report for more details).
- **semi-supervised_clustering.ipynb** : This notebook aims at clustering the dataset using a classifier that is trained on a small portion of the data. It uses a Residual Network as a classifier. After being trained, it is used to predict the sensitive attribute of the training data and allocates each sample to its corresponding cluster based on its prediction and the sample's class. This notebook can be run sequencially, cell by cell. (see report for more details).
- **K-means (basket_volley).ipynb** : This notebook explores the basket_volley dataset and investigates new approaches, it can be ignored.
- **K-means (doctor_nurse).ipynb** : This notebook explores the doctor_nurse dataset, which is not talked about in this project. This notebook can be ignored.

### Logistic Regression

This folder deals with the Adult tabular dataset described above. It contains scripts that train and measure the performance of baselines as well as our reweigthing algorithms (see report for description). The main scripts are listed here, as well as commands to run them:

- **main.py** : This is the main script of the folder. Its behavior will depend on the arguments passed.

```
python3 main.py -label_column={LABEL_COL} -protect_columns={PROTECT_COLS} -mode={MODE} -start_epoch={START_EPO
```

where:

1. label_column=<label_column>: the name of the target column
2. protect_columns=<protect_columns> (separated by a comma, no space)
    - gender - male vs female (protected)
    - race_White - white vs non-white (protected)
3. mode=<mode>
    - 0: Model is trained on bias dataset as it is, no reweighting
    - 1: Model is trained on customed dataset, where each sample is reweighted as to have the same number of minority and majority samples per class (only works when there is one protected column)
    - 2: Model is trained on customed dataset, where weights of each cluster/sample is dynamically updated
4. update=<update> (This parameter is only relevant when in MODE 2)
    - cluster: each cluster has a weight
    - sample: each sample has a weight
5. weights_init = <weights_init> (This parameter is only relevant in MODE 2)
    - 0: the cluster/sample weights are initialized with unit weight
    - 1: the cluster/sample weights are initialized with weights from MODE 1 (only works when there is one protected column)
6. start_epoch=<start_epoch> : the epoch to start from if the model has already been trained
7. num_epoch=<num_epoch>
8. id=<id>: the id of the trained model
9. num_trials=<num_trials>: number of times to repeat the training process
10. num_proxies= <num_proxies>: number of features to remove that are most correlated with the label
11. file_path=<file_path>: the file path of the preprocessed data
12. verbose=<verbose>
13. lr=<lr>: the learning rate
14. update_lr=<update_lr> : the learing rate of the weight updates
15. batch_size=
16. . balance=<balance>
    - 0: The training set and test set is not rebalanced in any way
    - 1: The training set is rebalanced in terms of labels and the test set is rebalanced in terms of labels and groups/subgroups

When the training procedure ends, the checkpoint as well as some evaluation statistics will be saved at the path:
`./Case_{MODE + 1}/checkpoints/model_ep_{START_EPOCH + NUM_EPOCH}/Run_{ID}/ .`

- **base_rate_generator.py** : This first splits the dataset into the majority and minority groups, and trains a logistic regression model on each of them. It then records the difference between the performances of the two classifiers. See details about the base rate analysis in the report.

```
python3 base_rate_generator.py -label_column={LABEL_COL} -protect_columns={PROTECT_COLS} -num_epoch={NUM_EPOCH
```

where

1. label_column=<label_column>: the name of the target column
2. protect_columns=<protect_columns> (separated by a comma, no space)
   - gender - male vs female (protected)
   - race_White - white vs non-white (protected)
3. num_epoch=<num_epoch>
4. id=<id>: the id of the trained model
5. num_trials=<num_trials>: number of times to repeat the training process
6. num_proxies= <num_proxies>: number of features to remove that are most correlated with the label
7. batch_size= <batch_size>: controls the batch size of the classifiers.
8. keep=
   - The proportion to keep when filtering the majority and minority groups (must be ]0,1])
9. filter_maj --filter_min
   - 1: filters the group to improve model predictions
   - 0: does not filter the group
   - -1: filters the group to worsen model predictions

When the training procedure ends, the checkpoint as well as some evaluation statistics will be saved at the path: `./base_rate_models/checkpoints/model_ep_{NUM_EPOCH}/Run_{ID}/ .`

- The other scripts of the folder are briefly mentioned below:

| File | Description |
|---|---|
| adversial_model.py | Baseline model, see: [1]. |
| calibrated_eq_odds_postprocessing.py | Baseline model: see [2] |
| disparate_impact_remover.py | Baseline model, see [3] |
| reject_option_classifier.py | Baseline mode, see [4] |
| evaluate.py | Evaluates a trained model against multiple fairness metrics |
| fairness_metrics.py | Contains the fairness metrics defined in the report |
| load_dataset.py | Preprocesses the dataset and prepares it for training |
| logistic_regression_model.py | Contains the model defintion, the training and evaluation methods |

[1] Brian Hu Zhang, Blake Lemoine, Margaret Mitchell. *Mitigating Unwanted Biases with Adversarial Learning*, Stanford University, Google Mountain View, CA (2018)

[2] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, andK. Q. Weinberger, *On Fairness and Calibration*, Conference on Neural Information Processing Systems, 2017

[3] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, *Certifying and removing disparate impact*, ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015.

[4] [10] F. Kamiran, A. Karim, and X. Zhang, *Decision Theory for Discrimination-Aware Classification*, IEEE International Conference on Data Mining, 2012.

## Resnet

This folder deals with the basket_volley image dataset described above. It contains scripts that train and measure the performance of baselines as well as our reweghting algorithms. Note that for images, the reweighting model is a

*Residual Neural Network* model, where we freeze all layers except from the last two, which we train on our dataset. The main scripts are listed here, as well as commands to run them:

- **Case_1+2.py** : This script takes care of the two following cases:
    - The resnet is trained on the dataset and its performance (accuracy vs fairness trade-off) is computed.
    - The resnet is trained on a reweighted dataset, where the minority class is reweighted by an input weight defined by the user

```
python3 Case_1+2.py -w_protected={W_PROTECTED} -bias={BIAS} -val_mode={VAL_MODE} -start_epoch={START_EPOCH
```

where:

1. w_protected=<w_protected>: weight with which to reweight every sample of the minority class
2. bias=<bias>: the version of the dataset to use, i.e. bias ranges from 0.5 to 0.8 and represents the ratio of `majority : minority` of each class.
3. start_epoch=<start_epoch> : the epoch to start from if the model has already been trained
4. num_epoch=<num_epoch>
5. val_mode=<val_mode>: whether to train the model in validation mode
6. id=<id>: the id of the trained model
7. num_trials=<num_trials>: number of times to repeat the training process
8. visom=<visdom>: boolean that determines whether to plot the training process of the model (visdom diplays the plots on an external server, see documentation: https://github.com/fossasia/visdom)

When the training procedure ends, the checkpoint as well as some evaluation statistics will be saved at the path:
`./("Case_2/" if W_PROTECTED != 1 else "Case_1/") + "checkpoints/" + ( "w_val" if VAL_MODE else "w.o_val") + f"/Bias_{BIAS}/model_ep_{START_EPOCH + NUM_EPOCH}/Run_{ID}/ .`

- **data_reweghting.py** : This scripts trains and evaluates our algorithms on the image dataset.

```
python3 data_reweighting.py -w_protected={W_PROTECTED} -bias={BIAS} -val_mode={VAL_MODE} -start_epoch={START_E
```

where:

1. w_protected=<w_protected>: weight with which to reweight every sample of the minority class
2. bias=<bias>: the version of the dataset to use, i.e. bias ranges from 0.5 to 0.8 and represents the ratio of `majority : minority` of each class
3. label_column=<label_column>: the name of the target column
4. start_epoch=<start_epoch> : the epoch to start from if the model has already been trained
5. num_epoch=<num_epoch>
6. val_mode=<val_mode>: whether to train the model in validation mode
7. id=<id>: the id of the trained model
8. num_trials=<num_trials>: number of times to repeat the training process
9. update=<update>: this argument decides which algorithm to use (see report for more detailed explainations):
    - cluster: each cluster has a weight
    - sample: each sample has a weight
    - individual: each sample is treated as an independent individuals
10. update_lr=<update_lr>: the weight update learning rate of the algorithm
11. clusters=<clusters>: this parameter should be set when using customized clusters. It should be the name of a python dictionary mapping each sample to its cluster

When the training procedure ends, the checkpoint as well as some evaluation statistics will be saved at the path:
`./"Reweighting/checkpoints/" + ("cluster_update/" if UPDATE == "cluster" else ("sample_update/" if UPDATE == "sample" else "individual_update/")) + ("w_val" if VAL_MODE else "w.o_val") + f"/Bias_{BIAS}/model_ep_{START_EPOCH + NUM_EPOCH}/Run_{ID}/ .`

- The other scripts of the folder are briefly mentioned below:

| File | Description |
|---|---|
| Case_3.py | Trains the model on a randomized reweighted dataset, this class is depreciated. |
| fairness_metrics.py | Contains the fairness metrics defined in the report |
| my\ImageFolder.py | Preprocesses the datastet, splits it in terms of clusters and prepares it for training |
| model.py | Contains the Resnet as well as methods to train and evaluate it |
| Result.py | Evaluates a trained model against multiple fairness metrics |