NTNU
Norwegian University of
Science and Technology

Master's thesis

2020

Gunnar Strand Jacobsen

Master's thesis

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Gunnar Strand Jacobsen

An Exploration of Bias and Fairness in Algorithmic Decision-Making Systems

June 2020

NTNU
Norwegian University of
Science and Technology

An Exploration of Bias and Fairness in Algorithmic Decision-Making Systems

Gunnar Strand Jacobsen

Informatics, Artificial Intelligence
Submission date: June 2020
Supervisor: Pinar Øzturk

Norwegian University of Science and Technology
Department of Computer Science

Abstract

Algorithmic decision-making systems assist, or sometimes even replace, human decision-makers in high impact settings both in the public and private sectors. The systems make decisions that significantly affect peoples' lives, such as access to credit, college admission, employment, medical treatment, or judicial sentencing. Thus it is clear that ethical aspects such as the fairness of these systems are of great importance. Unfortunately, algorithmic decision-making systems are known to be plagued by biases that can make their decisions discriminatory or unfair towards population subgroups. In response to this, "fair machine learning" has emerged as a new field of study, with the overarching goal to mitigate bias unintentionally incorporated into algorithms. This field is contributed to by researchers from diverse disciplines such as law, ethics, philosophy, computer science, statistics, machine learning, and social sciences, and these disciplines often lack common terminology, making fair ML literature scattered and sometimes confusing. Few fair ML papers attempt to present a comprehensive overview of common types of bias that can enter predictive ML systems. In this thesis, we fill a gap in previous research by doing just that. Furthermore, we review literature that shows that approaches intended to enhance the fairness of algorithmic decision-making systems tend to reduce prediction accuracy, forcing engineers and decision-makers to make trade-offs between accuracy and fairness. We explore these trade-offs, using multi-objective optimization via the genetic algorithm NSGA-II. Our findings show that there is a clear accuracy penalty for improving fairness via common bias mitigation tools, but also that sometimes, very large increases in fairness can be achieved at a relatively low accuracy cost.

Preface

This master thesis was written during the fall of 2019 / spring of 2020 at the Norwegian University of Science and Technology (NTNU), Faculty of Information Technology and Electrical Engineering, Department of Computer Science, as part of a Master of Science-degree in Informatics, Artificial Intelligence.

I would like to thank fellow student Pernille Johnsen, who have been investigating the same problems in parallel with me, for her invaluable help with the experiments, as well as for co-operation and sharing of ideas during the course of the project. I would also like to thank my supervisor Pinar Øzturk for all our great discussions, for having faith in me, and for leading me safely through the entire process of writing the masters.

Gunnar Strand Jacobsen

Trondheim, June 22, 2020

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Artificial intelligence (AI) is no longer a purely academic project. Thanks to advances in machine learning (ML), and the availability of massive amounts of data, AI systems now affect our everyday life. Examples of commonly encountered AI technologies include smart assistants in phones, spam filtering, personalized social media feeds, face recognition logins, and product recommendations in online shopping, to name a few. Perhaps less well known, but even more impactful to individuals and society, are the algorithmic decision-making systems (ADS) that assist or replace human decision-makers in many high stakes settings. These systems are used both in the private and public sectors. They rely on the analysis of personal data to infer correlations, based on which they make predictions that significantly affect peoples' lives. Because the impact of the predictions can be substantial, ethical aspects such as the fairness of algorithmic decision-making systems are of paramount importance.

Unfortunately, there are numerous examples that algorithmic decision-making systems are plagued by bias, such that they make decisions that are discriminatory or unfair towards population subgroups. Furthermore, research that we will review in this paper shows that approaches intended to enhance the fairness of the systems tend to reduce their prediction accuracy, forcing engineers and decision-makers to make trade-offs between accuracy and fairness.

In this thesis, we will investigate the biases that cause unfairness in algorithmic decision-making systems, as well as explore fairness-accuracy trade-offs in bias mitigation. Section 1.1 presents some background and motivation for the thesis, section 1.2 places algorithmic fairness within the broader field of AI ethics, section 1.3 contains our research questions, and section 1.4 lays out the thesis structure.

1.1 Background and motivation

Algorithmic decision-making systems are used in several socially impactful settings. They process credit applications [1], [2], assist in bail and probation proceedings [3], [4], screen job candidates [5], [6], predict locations of future crimes [7], [8], perform medical diagnosis based on chest X-rays [9], [10], decide who gets access to health care [11], and evaluate immigrant applications [12], to name a few.

Using algorithms for decision making may provide several benefits. In some areas, like screening X-ray images as part of medical diagnosis, patient outcomes may suffer due to shortages of radiologists [13], making algorithmic systems an attractive alternative. Algorithms make quicker and cheaper decisions than humans. They also make predictions based on datasets that are too large for humans to handle efficiently. Provided enough training data, they can often generate equally good decisions as human experts, e.g., demonstrate radiologist-level accuracy in diagnostic classification [9].

Furthermore, ADS may reduce the arbitrariness that is present when humans make decisions. It has been found, for example, that human judges sometimes judge similar defendants differently, that some judges are stricter than others [14], and that the decisions of human judges are affected by how hungry they are [15]. ADS will trivially avoid all those problems, acting reliably at all times.

Human decision making can be unintentionally swayed by cognitive biases or prejudice on the part of the decision-maker. Naively, one might assume that because automatic systems hold no prejudices against any group, all decisions they make will be purely objective. This assumption, though, does not to hold in real life. It has been known for almost 25 years that algorithmic systems and datasets can be biased [16], and later research has shown that algorithmic decision-making systems have the potential to reproduce or amplify human bias, or even introduce new bias [17][18]. Bias may cause the ADS to exhibit disparities in performance across protected subgroups, leading to different subgroups receiving differing treatment, thereby producing discriminatory or unfair outcomes for people with specific social or demographic traits. Table 1.1 shows some examples of algorithmic decision-making systems with fairness issues.

Area	**Fairness issues**
Public health care	A commercial algorithm from UnitedHealth Group, widely used by the US health care system to guide decisions about which patients should receive extra health care, has been shown to be racially biased. At a given risk score, black patients are considerably sicker than white patients, leading to unequal access to medical care for equally sick black and white patients. The bias is thought to occur because the algorithm uses health care costs as a proxy for health needs. Traditionally, more money is spent on white patients with the same level of need, so the algorithm falsely determines that black patients are healthier than equally sick white patients [19].
Medical diagnosis	A May 2020 paper has found that gender imbalances in X-ray image datasets lead to biased ADS for medical diagnosis [20]. Likely reasons are actual physiological differences in men and women, or that men and women tend to seek medical help and get their X-rays taken at different stages in the progression of their disease, so that an algorithm trained on mostly male X-ray images will not generalize well to women.
Recidivism prediction	The COMPAS system, an automated system used by some US courts for predicting the likelihood that a prisoner will commit a crime if released, has been shown to output higher false positives rates for black persons and higher false negative rates for white persons [3]. The system is not given explicit information about race, but derives information about race from proxies like education and ZIP code [21].
Image classification	It is a well-known problem in image classification that algorithms perform more poorly on black persons than white persons. A notorious example is when Google Photos labeled black people as gorillas [22]. Part of the reason for this is that common datasets used for training the algorithms contain too few images of people of color.
Credit approval	One famous example is the gender bias exhibited by the Apple credit card issued in the fall of 2019, which used an algorithm from Goldman Sachs to determine credit limits. The algorithm gave Apple co-founder Steve Wozniak a ten times higher credit limit than that of his wife, despite the two sharing all assets and accounts [23].

Table 1.1: Examples of AI Systems with Known Fairness Issues

In response to this, "fair machine learning" has emerged as a new field of study, with the overarching goal to mitigate bias unintentionally incorporated into algorithms. This field is contributed to by researchers from diverse disciplines such as law, ethics, philosophy, computer science, statistics, machine learning, and social sciences, and these disciplines often lack common terminology, making fair ML literature scattered and sometimes confusing. To our knowledge, few papers attempt to present a comprehensive overview of common types of bias that can enter predictive ML systems. In this thesis, we fill a gap in previous research by doing just that.

Developing a good understanding of bias and how it can affect outcomes of algorithmic decisions is, of course, vital for engineers so that they can develop fairer decision-making systems. But it is also crucial for those who use or oversee these systems. Because, while unfair outcomes are particularly worrisome when the ADS acts autonomously [24], research shows that even with a human in the loop, biased decisions might slip through unnoticed. The reason for that is that most algorithms are so complex and inscrutable that it is hard for humans to detect possible bias in their decisions [24]. Furthermore, if the human does not have a good understanding of the potential bias that may be present in the algorithm, there is a risk of the human trusting the system too much [25], thereby diminishing the value of human oversight.

Many sub-fields of machine learning consider algorithmic fairness, e.g., recommender systems [26], or text mining [27], but in this thesis, we will concentrate only on fairness in algorithmic decision-making systems. How to mitigate bias and unfairness in ADS is a very active area of research in fair ML, and there are several approaches for this, some of which we will describe in section 2.6. As we will see in our review of related work, it is well established that enhancing algorithmic fairness typically comes at the cost of reducing prediction accuracy [28]–[35]. In other words, there is a trade-off between fairness and accuracy. We will look for state-of-the-art methods or frameworks for evaluating those trade-offs, try to reproduce selected results, and then attempt to verify that the methods work by applying them on new problem settings. If we are successful, we will have made contributions toward helping decision-makers choose the most appropriate levels of fairness for their particular use cases and problem settings.

1.2 Fair ML Within the Broader Field of AI Ethics

As a result of increased awareness around potential risks, harms, and disruptive effects of AI technologies, there has been growing interest in the ethics of AI. Initiatives and institutes have been founded to study these issues, and at least 63 guidelines for the ethical development, deployment, and governance of AI have been published [36]. Ethics in this context covers a variety of questions of right and wrong, bias and distortion, power and abuse. It investigates individual and societal harms from *"the misuse, abuse, poor design, or negative unintended consequences of AI systems"* [17].

While some researchers work on challenges we may face if we succeed in creating artificial general intelligence (AGI) [37], [38], others concentrate on the potential for harm in technologies we already do possess. AI researcher Pedro Domingos has said: *"People worry that computers will get too smart and take over the world, but the real problem is that they're too stupid and they've already taken over the world"* [39]. The statement refers to the fact that machine learning systems make decisions that significantly impact peoples' lives, solely based on correlations in data, without any understanding or true intelligence. Such systems are susceptible to make ethical mistakes.

The challenges discussed in this thesis falls under the umbrella of fair ML, which first and foremost deals with unfairness and bias in algorithmic decision-making. Fair ML has been singled out by many AI ethics organizations as an area that needs immediate attention, and is an active research area, with several conferences and workshops dedicated to it, e.g. FAT/ML, ACM FAT and FairWare.

1.3 Goals and Research Questions

The overall goal of this thesis is to map the types of algorithmic bias that may cause unfair outcomes for protected groups, and to explore trade-offs between accuracy and fairness, in algorithmic decision making systems.

Towards that end, we formulate the following research questions:

Research question 1 <u>What biases in algorithms and datasets cause unfair outcomes in machine learning systems?</u>

Research question 2 <u>What are the state-of-the-art methods or frameworks for investigating trade-offs between fairness and accuracy in machine learning classification systems?</u>

Research question 3 <u>Can we apply the method(s) or framework(s) identified in RQ2 on new problem settings, and what conclusions can we draw from that?</u>

1.4 Thesis Structure

The thesis contains six chapters. The first provides background and motivation for our work and presents our goals and research questions. The second chapter contains the background theory that is necessary in order to follow the subsequent discussions. Chapter three contains literature reviews of related work, as well as lays out our approach to bias. Chapter four explains the approach to method and experiments, and chapter 5 contains the experimental results. Finally, in chapter 6, we summarize, evaluate, and wrap up with conclusions, as well as provide some suggestions for future work.

Chapter 2

Background Theory

This section presents the theoretical background of this Master's thesis. Our aim is for the reader to gain the necessary background knowledge for understanding the subsequent discussions. We will also define some key terms. First, section 2.1 contains a brief overview of machine learning and algorithmic decision-making systems, and then section 2.1.2 introduces multi-objective optimization, Pareto optimality, and Pareto fronts. Section 2.1.3 presents genetic algorithms, and section 2.1.4 goes in-depth on the topic of fairness. Sections 2.1.5 and 2.1.6 covers bias and bias mitigation methods, respectively.

2.1 Machine Learning and Algorithmic Decision-making Systems

Computer systems can make sense of large quantities of data through machine learning, allowing them to complete certain tasks quickly and at great scale, and to solve problems that would not be cost-effective - and in some cases not even feasible - to solve by manual programming. Two examples of the latter are playing advanced games better than any human [40], or detecting skin cancer with more accuracy than dermatologists [41].

The traditional machine learning pipeline consists of the following five steps: (1) ingesting data, (2) preparing data, (3) feature extraction, (4) training models, and (5) serving predictions. Figure 2.1 illustrates this pipeline. Because the traditional machine learning pipeline does not deal with unfairness, it is necessary to include additional strategies. We will discuss that in greater detail in later chapters.

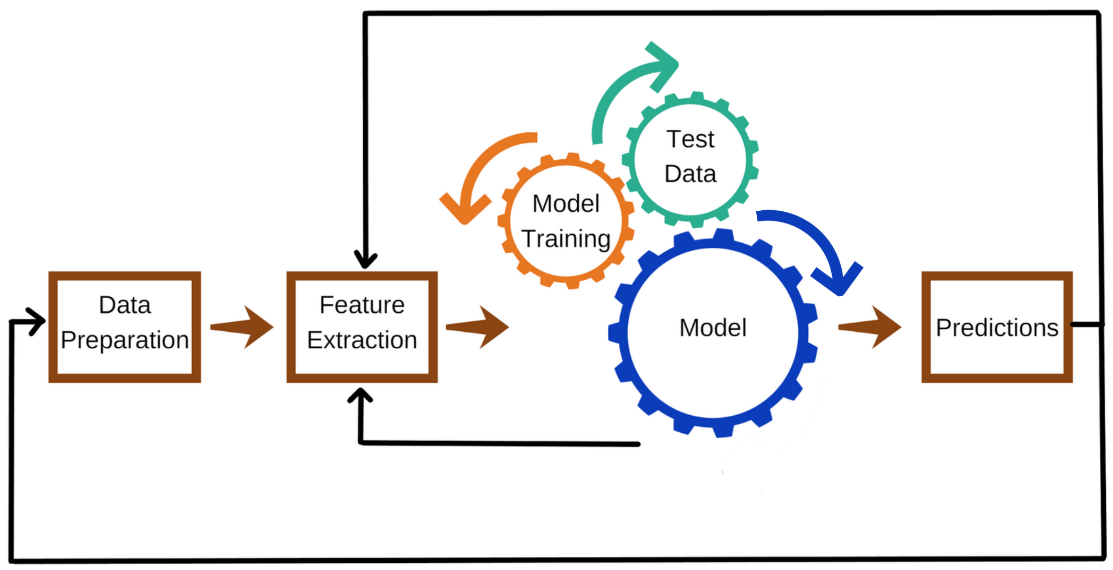


Figure 2.1: *A Standard Machine Learning Pipeline*

Although there are many types of machine learning, the type that is most relevant to our paper is *supervised learning*, in which the algorithm observes some example input-output pairs and then learns a function that maps from input to output. Specifically, the task of supervised learning is this:

> Given a set of n example input-output pairs $(x_1, y_1),\dots,(x_n, y_n)$, where the true outputs y_i are known, but the function $y=f(x)$ that generated them is not known, discover a function $h(x)$ that approximates $f(x)$

Learning a discrete-valued function is called *classification*, and learning a continuous function is called *regression*. When there are only two possible values for the outcome, it is called *binary classification*. When classifiers are used for decision making or predictive analysis, we will refer to them as *algorithmic decision-making systems*.

The data set containing the input-output pairs used to develop h is called a *training set*. After training, the algorithm's performance can be tested on a set of previously unseen data, and this set of data is called a *test set*.

We are mostly interested in classification problems in this paper, and the algorithms for solving them are called *classifiers*. Algorithms usually used for regression can also be modified to work for binary classification problems, by deciding on a *classification threshold*, or *decision threshold,* for separating positives from negatives. For instance, in a credit approval system, predicted credit scores above the threshold can be labeled as belonging to the positive class, and predicted credit scores below the threshold can be labeled as belonging to the negative class.

Machine learning algorithms for classification are evaluated based on their *accuracy.* Informally, accuracy is the fraction of predictions that the algorithm got right. Formally:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

Outcomes of classifiers can be divided into *true positives (TP)*, which are outcomes where the classifier correctly predicts the positive class, *true negatives (TN)*, which are outcomes where the model correctly predicts the negative class, *false positives (FP)*, which are outcomes where the classifier incorrectly predicts the positive class, and *false negatives (FN)*, which are outcomes where the classifier incorrectly predicts the negative class. This allows us to create a better accuracy formula for binary classification:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

In many settings, accuracy alone is not a very valuable measure of the classifier's performance. For instance, an algorithm for detecting a rare form of cancer that usually turns up in only one of a thousand tumors will have an accuracy of 99,9% if it simply classifies all tumors as benign. Yet, that will be a despicable performance, leaving 100% of all malign tumors undiagnosed.

To better evaluate the classifier's true performance, one can look at *precision* and *recall.* Precision attempts to find what proportion of positive identifications was actually correct:

$$Precision = \frac{TP}{TP + FP}$$

Recall attempts to find what proportion of actual positives was identified correctly:

$$Recall = \frac{TP}{TP + FN}$$

In our example above, with the classifier identifying all tumors as benign, the classifier would have a recall of 0, thus showing that it has horrible performance despite its 99,9% accuracy.

Recall is sometimes called *true positive rate (TPR),* and a *false positive rate (FPR)* is defined as:

$$FPR = \frac{FP}{FP + TN}$$

The final classifier performance measure we wish to mention is called Area under Curve (AUC), or Area Under the ROC Curve. The ROC curve is a graph showing the performance of a classification model at all classification thresholds, plotting the parameters TPR and FPR on its axis. AUC measures the two-dimensional area underneath the ROC curve, giving values from 0.0 (when the model predictions are 100% wrong), to 1.0 (when the model predictions are 100% correct). AUC is scale-invariant, and clasification-threshold invariant. In some cases it is a preferable performance measure to the measures mentioned above.

In the next section we will present a specific classifier, which we shall use in the experiments that attempts to answer research question 3.

2.1.1 Support Vector Machines

A Support Vector Machine (SVM) is a highly popular, supervised machine learning algorithm, capable of performing both linear or nonlinear classification, as well as regression, making them part of step (4) of the machine learning pipeline mentioned in the previous section. Support Vector Machines belong to the class of non-parametric methods, meaning they retain training examples and potentially need to store them all. In practice, however, they often only have to retain a small fraction of the number of examples. Thus they can be said to combine advantages of non-parametric and parametric models, by being resistant to overfitting, but still having the flexibility to represent complex functions. Another benefit of SVMs is that they generalize well. This is due to their method of constructing maximum margin separators; *decisions boundaries* with the largest possible distance to example points.

For linearly separable data, the SVM can perform *hard margin classification.* Figure 2.2 (a) shows an example, where the two classes in a dataset are separated by *support vectors* in such a way that there are no data points between the support vectors. The support vectors are separated in the middle by the decision boundary, and when the distance between the support vectors is maximized, the decision boundary is called an *optimal hyperplane.* Hard margin classification is sensitive to outliers. This problem can be handled by allowing margin violations, i.e. allowing data points to be on the "wrong" side of the support vectors. This is called soft margin classification. In soft margin classification it is still important to strike a good balance between the margin violations and the goal of maximizing margins, and this balance can be controlled through the C hyperparameter, a.k.a. the penalty parameter, a.k.a. the soft margin constant. It determines the cost of mis-classification, in the form of a penalty to apply to data points that end up on the wrong side of the decision boundary. A smaller C give a wider margin, but at the price of more margin violations, whereas a larger C will give fewer margin violations but smaller margin. Incidentally, a smaller C will also make the SVM less likely to overfit the data.

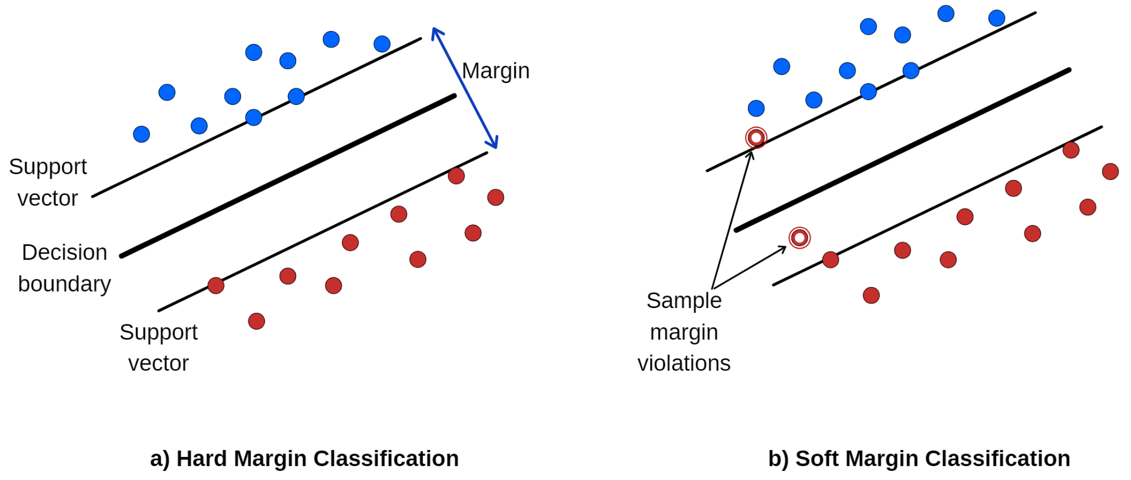


Figure 2.2: *Support Vector Machine Concepts*

We can adjust SVMs to work for datasets that are not linearly separable, by using a mathematical technique called *kernel functions*. Kernel functions specify the type of decision boundary that can be learned by the classifier, and the calculations needed to ensure this. Polynomial kernels and radial kernels are two examples of non-linear kernels. Non-linear kernels are more flexible, but comes at the cost of requiring more computational power, which might become a problem for large datasets. Therefore, the choice of kernel is an important decision when working with SVMs, and the choice of kernel affects what hyperparameters that need to be tuned (in addition to C).

In our thesis we will be performing soft margin classification with a Gaussian radial basis function (RBF) kernel, which has the following equation:

$$\phi_{\gamma}(\mathbf{x}, l) = \exp(-\mu \|\mathbf{x} - l\|^2), \mu > 0$$

where $\mathbf{x} = (x_1, x_2)$, l is the center from which the euclidian distance of $\mathbf{x}$ is measured, and γ is the kernel parameter. Along with C, the kernel parameter determines the performance of the SVM model, or more specifically, the relationship between the model's bias and variance. A large γ value leads to high bias and low variance, while a low γ value gives low bias and high variance. C works the other way; a large C gives low bias and high variance, and a small C gives higher bias and lower variance.

2.2 Multi-Optimization, Pareto Optimality and Pareto Fronts

Within mathematics, *optimization* means attempting to maximize or minimize some function relative to some set. Formally, given a possibly nonlinear and non-convex continuous function $f : \Phi \rightarrow \mathbb{R}$ from some set Φ to the real numbers, find an element $\mathbf{x}_0 \in \Phi$ such that $f(\mathbf{x}_0) \geq f(\mathbf{x})$ for all $\mathbf{x} \in \Phi$ (maximization) or such that $f(\mathbf{x}_0) \leq f(\mathbf{x})$ for all $\mathbf{x} \in \Phi$ (minimization). Common applications include finding minimal cost, maximal profits, minimal error, and so forth. The domain Φ of f is called the *search space,* while the elements of Φ are called *feasible solutions.* The function f is called the *objective* function, and is a *loss function* or *cost function* when the problem at hand is minimization, and a *utility function* or *fitness function* when the problem is maximization. Finding arbitrary local minima or maxima is usually relatively straightforward, but finding a global minimum or maximum can be very challenging.

When there is more than one objective function to be optimized simultaneously, the problem is called *multi-objective optimization (MOO)*. It is used in situations where optimal decisions need to be taken in the presence of trade-offs between two or more conflicting objectives. For non-trivial multi-objective optimization problems, it is impossible to find a single solution that simultaneously optimizes each objective. The objective functions are then said to be conflicting.

The concept of optimization is not relevant only in mathematics; it is used in engineering, economics, and many other disciplines. In engineering, an example of conflicting objectives may be the minimization of the production cost of some product, while maximizing its quality. Barring knowledge about which objective is more important, decisions will need to be taken in the presence of trade-offs between the two conflicting objectives. There will then exist a number of *Pareto optimal* solutions. A solution is Pareto optimal if none of its objectives can be improved without degrading at least one of the other objectives.

Formally: given a set of objective functions $f_i(\mathrm{x})$, $\mathrm{i} \in [1, k]$, and a multi-objective optimization problem formulated as

$$\min(f_1(x), f_2(x), \ldots, f_k(\mathrm{x}))$$

a solution ẋ *Pareto dominates* ẍ if the following two conditions hold:

1. $\forall\ \mathrm{i} \in [\{1,\ldots,k\}\ :\ f_i(\dot{\mathrm{x}}) \leq f_i(\ddot{\mathrm{x}})$
2. $\exists\ \mathrm{j} \in \{1,\ldots,k\}\ :\ f_j(\dot{\mathrm{x}}) < f_j(\ddot{\mathrm{x}})$

In other words: a solution ẋ Pareto dominates *a solution* ẍ if ẋ is at least as good as ẍ for all objectives, and ẋ is strictly better than ẍ for at least one objective.

It then follows that Pareto optimal solutions are the solutions that are not domintated by any other solution. For this reason they are also known as *nondominated* solutions.

The set of all Pareto optimal solutions for a given system is called the *Pareto front*. All Pareto optimal solutions are objectively equally good, so further decisions will rely upon the subjective preferences of a human decision-maker. Figure 2.3 shows an example Pareto front, for a problem in which two objectives; A and B, both need to be maximized. The dots on the Pareto front curve are the nondominated solutions, and the dots under the curve are the dominated solutions.

The classical approach for solving multi-objective optimization problems is to use methods such as the weighted sum method, lexicographic ordering methods, or scalarization methods. These use various strategies to convert the problem into a single-objective optimization problem. Some downsides of the classical approaches are that they will only deal with one solution per run, may miss some solutions, have difficulties dealing with discontinuous or concave Pareto fronts, and may require a priori information about preferences, ranking, or even weights of the objectives.

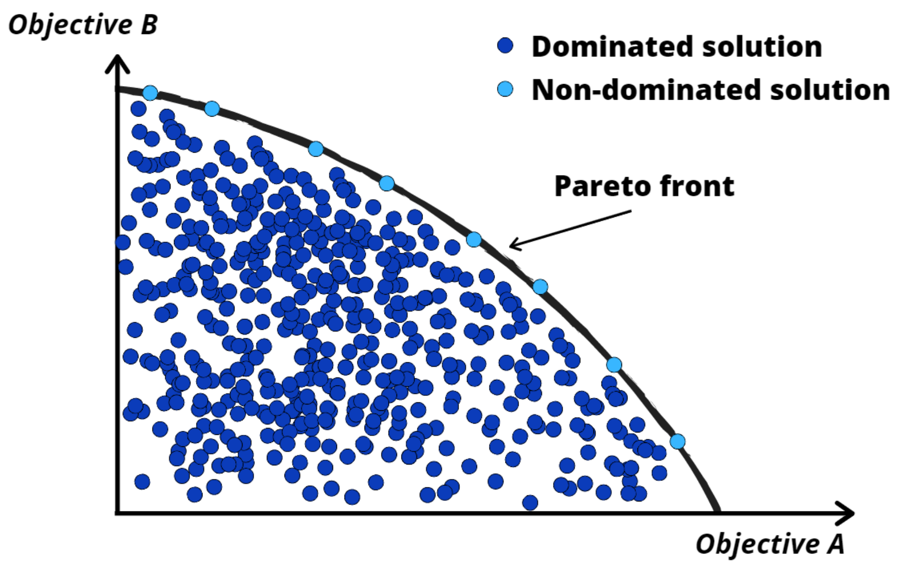


Figure 2.3: *Pareto front*

A popular category of methods for MOO problems that do not have these limitations are the bio-inspired, multi-objective *evolutionary algorithms*. Many different types exist, but we will look in particular at one of the more popular variants, called a *genetic algorithm (GA)*. Genetic algorithms offer certain benefits compared to traditional MOO approaches; the main ones for us are the ability to find several Pareto optimal solutions per run of the algorithm, and that they do not need a priori preference information about the objectives. The next section gives the necessary background on genetic algorithms.

2.3 Genetic Algorithms

Like we mentioned at the end of the previous section, a genetic algorithm is a subclass of evolutionary algorithms. Evolutionary algorithms are algorithms based on Darwin's theory of evolution. Evolution is the change in the genetic makeup of populations over time, and Charles Darwin suggested that *natural selection* is the primary mechanism behind evolution. Natural selection says that organisms with the best-suited characteristics for their environment survive and pass on their genetic traits in increasing numbers to successive generations. In contrast, less adapted organisms reproduce at a lower rate or are eliminated from the population. For natural selection to occur, there must be variation within populations, and competition between organisms for resources. An organisms' ability to survive and reproduce in its environment is called its *fitness*, and the principle of natural selection is often branded "survival of the fittest".

Members of a population produce offspring through cross-breeding, thus creating a new *generation* of individuals. The offspring inherits traits from their parents through *genes*. Genes are stored inside *chromosomes*, and each gene has a special place within a chromosome, called its *locus.*

Variation in the population stems from the introduction of new genes via random changes called mutations, and via reshuffling of existing genes. Because only the fittest tend to survive and reproduce, it is expected that the very fittest individuals will be found in the latest generations.

Genetic algorithms are inspired by the theory of evolution by being stochastic, population based algorithms that rely on biologically inspired operators such as mutation, crossover, and selection. Although not mathematically guaranteed to find the optimal solution, they often find sufficiently good solutions for certain optimization problems. Briefly, the idea is to first generate an initial population of candidate solutions to the problem at hand, and then iteratively generate new generations of solutions by applying stochastic operators such as mutation, crossover, and selection on the previous population. The initial population is usually seeded by randomly generated candidate solutions. Each candidate solution in the population is called an individual, and the solutions are encoded in chromosomes, which contain genes, which are in fixed positions called loci and have a value (*allele*).

There are many possible representations for the problem being solved. The chromosomes can be any data structure, but the typical example is an array of bits. Figure 2.4 shows an example, while also illustrating the concepts of genes and population.

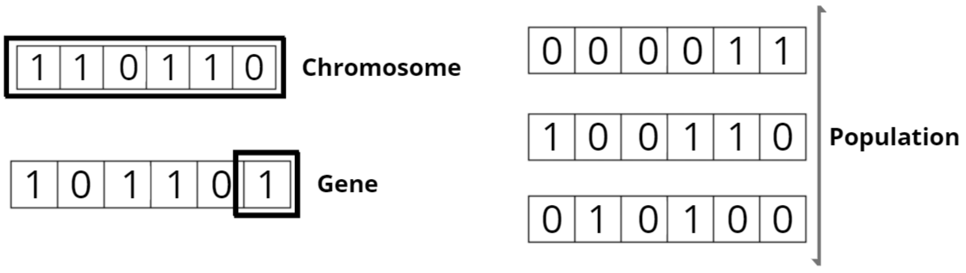


Figure 2.4: *Population, chromosomes and genes*

We can split the representation of a chromosome into two parts, *the genotype,* and *the phenotype.* The genotype is the encoding of the solution used by the evolutionary algorithms, while the phenotype is the encoding that represents the real world solution.

In nature, the fittest individuals are more likely to reproduce, and genetic algorithms emulate this by basing the selection probability for reproduction on a fitness function. The fitness function is usually the objective function in the optimization problem being solved. The fitness function outputs a single real-valued fitness score for each individual in the population, and can be looked upon as a measure of the quality of the solution that the individual represents. To calculate fitness scores, the chromosome genotype must generally be decoded into a phenotype that can be used to calculate these scores.

A certain number of individuals are randomly selected for reproduction in each generation, with a probability for selection that increases with increasing fitness function scores. Individuals may be selected more than once, and even the worst individuals in the population have a non-zero chance of being selected.

The next generation is generated from those selected in the previous generation through a combination of genetic operators: recombination (usually called crossover), and mutation. Figure 2.5 visualizes crossover, and figure 2.6 gives two examples of mutation operators: swap, and flip. Several other mutation operators exist, but those are the ones we will use in this thesis. Crossover and mutation processes ensure that the next generation population is different from the previous. Over time, the average fitness of the population will usually increase by this procedure, due to the tendency to select the fittest individuals for reproduction.

New generations are produced until some termination criterion is met. For instance, when a good enough solution is found, or because a fixed number of generations is reached, or because the fitness of the best solution has reached a plateau for a number of generations.

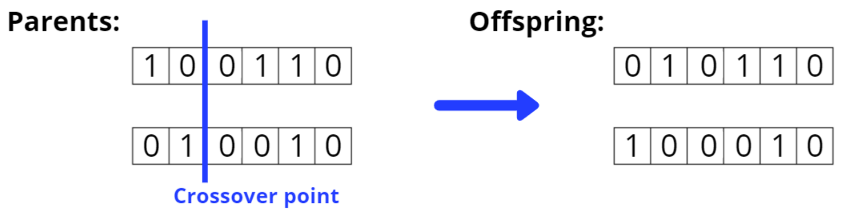


Figure 2.5: *Crossover*

For each pair of parents to be mated, a crossover point is chosen at random, and offspring are created by letting each child get its genes from one of the parents until the crossover point is reached, and from the other parent for the rest of the genes.

Summing up, the genetic algorithms contain four overall steps: (i) generating an initial population, (ii) calculate fitness of each individual, (iii) selection, (iv) crossover and genetic operators, and (iv) termination.

Sometimes genetic algorithms use *elitism*, which is the practice of copying a small portion of the fittest candidates, unchanged, into the next generation. Elitism can speed up the performance of the GA significantly.

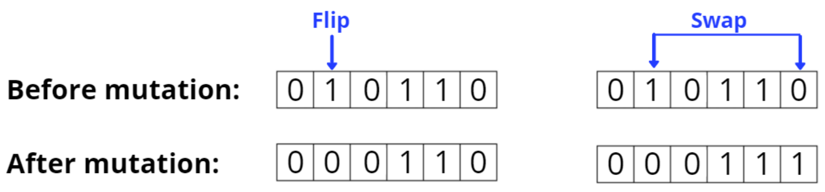


Figure 2.6: *Mutation operators*

Flip mutation: a gene is randomly selected, and flipped.

Swap mutation: Two randomly selected genes are swapped.

For multi-objective optimization problems, the solutions with the best fitness scores will make up a Pareto front. For each solution in the Pareto front there exists no other solution that is at least as good in all scores, and simultaneously better in at least one score. The *crowding distance* of a solution is the average distance to its two neighboring solutions. It can be said to be a measure of closeness in performance to other solutions. By biasing the selection of solutions towards those with greater crowding distances, one can ensure more diversity in solutions. Figure 2.7 shows how this works when selecting 7 out of fourteen solutions in a Pareto front.

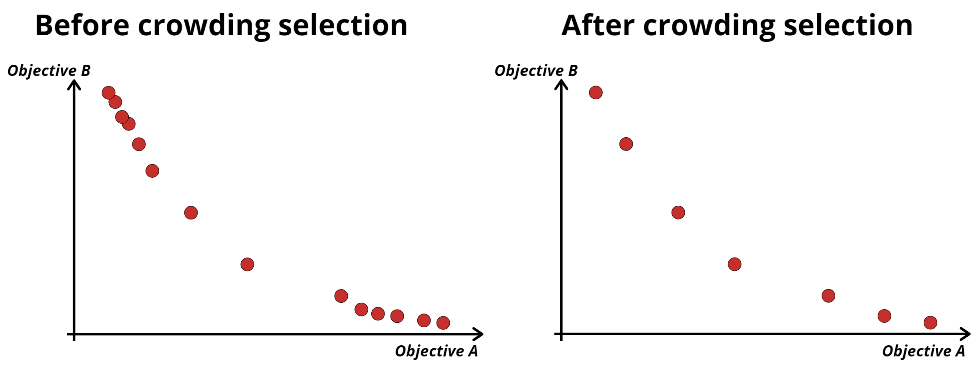


Figure 2.7: *Crowding selection*

Genetic algorithms are well suited to explore Pareto-dominance relationships in MOOs, by being able to evaluate a large number of classifier parametrizations, evaluate their performance on the objectives, and propagate non-dominated solutions.

2.4 Fairness and discrimination

Discrimination in a social sense of the word is the prejudiced treatment of people based on perceived membership in certain classes, groups, or categories, often called *protected classes*. The attribute that defines a protected class is called a *sensitive attribute*. Gender, race, religion, disability, or age are examples of sensitive attributes. Intentional discrimination explicitly based on sensitive attributes is called *disparate treatment* [29] and is prohibited by law in most countries [42]. Sensitive attributes are not universal; they are context and application-specific [43]. In cases where one can prove that a specific attribute is relevant to the outcome, it may be permissive for the algorithm to use that attribute, even though the same attribute might be considered sensitive in another setting. For example, governments can refuse blind people to become truck drivers, because being blind is relevant to the task [44]. Similarly, car insurance agencies are allowed to charge different prices to customers based on gender and age, because they can demonstrate that the risk of accidents is higher for young males than for older females.

In some cases, one may have *unintended discrimination*, where different groups receive different outcomes or treatment even though their protected class membership was not explicitly considered in the decision process. This is called *disparate impact* [29], and is the type of discrimination that most frequently occurs in biased algorithms.

Fairness is a concept that deals with impartial and just treatment of people. Often it is used in the context of the distribution of goods or rights. Different cultures have different notions and intuitions about what fairness is, but in most democratic societies, it is seen as an equal distributions of life chances. That regardless of their initial position in society, people with equal gifts should have equal opportunities.

Unfairness, then, is to limit people's life chances based not on merit or their contribution to society, but instead based on sensitive attributes like gender or race, or unrelated choices like where they live. An example of such unfairness might be to deny someone access to a loan based solely on their gender.

For machine learning purposes, one needs precise metrics for fairness, and many different metrics have been proposed. A basic and naive idea is called *fairness through unawareness/blindness,* which says that an algorithm is *fair* so long as it does not explicitly consider sensitive attributes in the decision process [45]. However, already in 2008, Pedreshi et al. showed that for sufficiently rich datasets, there might exist redundant encodings, i.e., highly correlated features that are proxies of the sensitive attribute, allowing the algorithm to predict unknown sensitive attributes from other features [46]. Another fairness concept that has been studied, most notably by Dwork et al., is called *individual fairness* [47], which is the idea that similar individuals should be treated similarly. The majority of *fair ML* research, though, studies *statistical fairness* [1], [21], [31], [48]. These are closely tied to the concept of discrimination [49]. Statistical fairness requires the equalization of a particular statistical metric across groups. Among the more notable such metrics is *demographic parity:* the idea that membership in a protected class should have no correlation with the decision. Research by Srivastava et al. indicates that this metric closely matches lay people's perception of fairness [49]. Unfortunately, Dwork et al. has shown that it does not ensure fairness, and that it disallows perfect prediction in cases where the outcome actually *is* correlated with the sensitive attribute [47].

Some fairness metrics are mutually incompatible, in the sense that they cannot hold simultaneously [21]. This can be illustrated by the so-called COMPAS debate. COMPAS is a probation risk assessment system used by courts in some states in the USA, which was the subject of a highly influential article in ProPublica by Julia Angwin, claiming that it is racially biased [3]. Angwin implicitly adopted *equality of false positive rates* as a fairness criterion in her article, showing that the COMPAS system had different false positive rates for white and black defendants. Among defendants who did not get rearrested, black defendants were twice as likely to be misclassified as high risk by the system [50]. In a rebuttal, Northpointe, the company behind COMPAS, has shown that the system satisfies equal *positive predictive values*.

Specifically, out of those classified as high risk, the proportion of rearrested people was approximately equal between races [50]. Hence, we can argue that the system is fair or unfair, depending on what fairness metric we use to evaluate it.

Srivastava et al. have pointed out that fairness is highly dependent on the context and societal domain in which the algorithm is used [49]. In light of that, developers need to be firmly aware of what fairness metric they want their system to satisfy, for the particular use case and context the system is intended for, and use this to guide the rest of the development process.

We will now present some well-known fairness metrics and their mathematical definitions (table 2.2).

Notations used in the table are:

- G: protected or sensitive attribute; G=1 means membership in the protected/unprivileged group, and G=0 means membership in the non-protected/ privilaged group.
- X: all additional attributes
- Y: the actual label / classification result
- S: predicted probability for a certain classification c, P(Y=c | G, X)
- d: predicted decision for the individual; d=1 means the individual received a positive classification, and d=0 means the individual received a negative classification
- E(…): expected value of...

Furthermore, when the wording “both groups” is used in the table, it refers to the protected group and the non-protected group.

Metric name	Definition	Reference
Statistical parity a.k.a equal acceptance rate a.k.a. demographic parity	Requires the positive outcome to be given at the same rate for both groups. $P(d=1 \mid G=1) = P(d=1 \mid G=0)$	Dwork et al (2012) [47]
Statistical parity difference	This is the difference in the probability of favorable outcomes between the protected and non-protected groups. $SP_{Diff} = P(d=1 \mid G=1) - P(d=1 \mid G=0)$	Bellamy et al. (2019) [43]
False negative error rate balance a.k.a. equal opportunity	Requires the same true positive rate in both groups. $P(d=0 \mid Y=1, G=1) = P(d=0 \mid Y=1, G=0)$	Hardt, Price, Srebro (2016) [1]
Equal opportunity Difference	Difference in true positive rates between protected and non-protected groups. $EO_{Diff} = P(d=0 \mid Y=1, G=1) - P(d=0 \mid Y=1, G=0)$	
Equalized odds a.k.a disparate mistreatment	Requires the same true positive rate and the same false positive rate in both groups. $P(d=1 \mid Y=i, G=1) = P(d=1 \mid Y=i, G=0), i \in \{0, 1\}$	Hardt, Price, Srebro (2016) [1]
Balance for the negative class	$E(S \mid Y=0, G=1) = E(S \mid Y=0, G=0)$	Kleinberg, Mullainathan, Raghavan (2017) [32]
Balance for the positive class	$E(S \mid Y=1, G=1) = E(S \mid Y=1, G=0)$	Kleinberg, Mullainathan, Raghavan (2017) [32]

Predictive parity	Both groups have equal positive predictive value (i.e. precision). $P(Y=1 \mid d=1, G=1) = P(Y=1 \mid d=1, G=0)$	Chouldechova (2016) [52]
Conditional use accuracy equality	Requires the same positive predictive value and the same negative predictive value in both groups. $(P(Y=1 \mid d=1, G=1) = P(Y=1 \mid d=1, G=0)) \land (P(Y=0 \mid d=0, G=1) = P(Y=0 \mid d=0, G=0))$	Berk et al (2017) [51]
False positive error rate balance a.k.a. predictive equality	Both groups have equal false positive rate. $P(d=1 \mid Y=0, G=1) = P(d=1 \mid Y=0, G=0)$	Chouldechova (2016) [52]
Overall accuracy equality	$P(d=Y, G=1) = P(d=Y, G=0)$	Berk et al (2017) [51]
Treatment equality	Both protected and unprotected groups have an equal ratio of FP and FN.	Berk et al (2017) [51]
Fairness through unawareness	No sensitive attributes are explicitly used in the decision process.	Kusner et al (2017) [53]
Disparate impact factor	This is the ratio in the probability of favorable outcomes between the protected and non-protected groups. $DI = P(d=1 \mid G=1) / P(d=1 \mid G=0)$	Feldman et al. (2015) [29]
Mean difference score	$MD = P(d=1 \mid G=1) - P(d=1 \mid G=0)$	Calders and Verwer (2010) [48]
Average odds difference	This is the average of difference in false positive rates and true positive rates between protected and non-protected groups.	Bellamy et al. (2019) [43]
Theil Index	Measures whether individuals are treated in a similar way. (Is the only individual fairness metric in this table)	Speicher et al. (2018) [54]

Table 2.2: A selection of common fairness metrics

2.5 Bias

In humans, bias is a tendency to have a preconceived or unreasoned positive or negative opinion of someone or something, usually in a way that is prejudicial, close-minded, or unfair. Human biases can impede fair decisions in many domains, both in the private and public sector.

In machine learning, bias is an overloaded word. It may refer to a technical bias in the algorithm, like the algorithm's inductive bias, or it may refer to a statistical bias affecting the accuracy of the outcome. An example of the latter is an algorithm's "tendency to consistently learn the same wrong thing" [55]. For example, if an algorithm for estimating credit scores consistently gives everyone a score that is a couple of points lower than their true risk, we say that it is *biased.* See figure 2.8 for an example of how bias works in the context of dart-throwing.

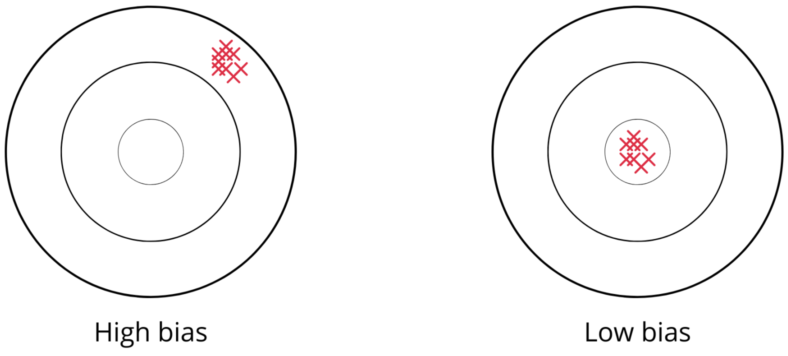


Figure 2.8: *Bias in dart throwing*

This type of bias affects all algorithms to some degree and is not the type of bias with which fair ML research concerns itself. However, if the algorithm systematically gives a poorer outcome for one subgroup than for another, and true demographic differences between the groups can not warrant this, the algorithm is *unfairly biased*. See figure 2.9 for a visualization. Unfair bias places privileged groups at a systematic advantage and unprivileged groups at a systematic disadvantage [43]. This form of bias has become a hot topic in machine learning in the last decade. Many research papers and news stories have revealed that some algorithms – even those used in high-impact public sectors like law enforcement – discriminate unfairly against minorities [3][56][57]. Despite this, these very same systems are increasingly being used in the public sector [58]. The systems are often bought from external vendors that developed them in isolation from users and use contexts [58], and they are often delivered without documentation that provides information about potential pitfalls or intended use cases of the system [59]. This, of course, exacerbates the potential for harm.

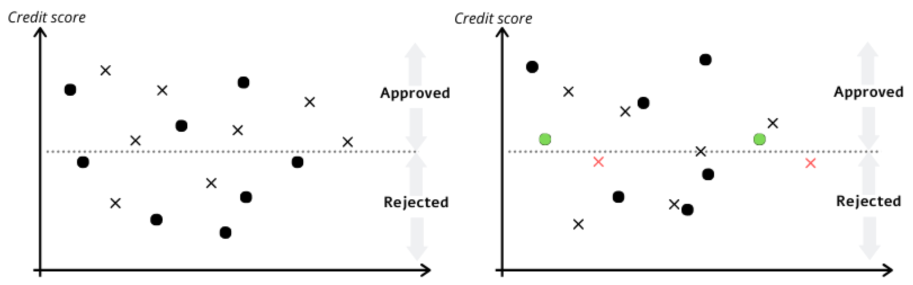


Figure 2.9: *Unfair bias in a credit approval algorithm.*

Left: *The true outcome.*

Right: *outcome of an unfairly biased algorithm. The accuracy for the algorithm as a whole is good, and the accuracy between groups (circles and crosses) are equal, but the algorithm is unfairly biased, systematically giving higher scores for circles, and lower scores for crosses, resulting in two false positives for the circles, and two false negatives for the crosses.*

So how can algorithms become biased? Machine learning algorithms learn from historical instances of a decision problem by picking up statistical patterns in data, so if that data reflects patterns of historical discrimination against some group, the algorithm is likely to reproduce or even amplify that discrimination [60]. Another way for bias to enter the algorithm, is if the data used to train or test it is not sufficiently representative of the populations about which it is drawing inferences. Even if the algorithm's overall accuracy is good, it might perform poorly for a minority group that was under-represented in the data that the algorithm was trained on [61]. In some cases, the algorithm itself may cause bias and discriminatory results [24].

A difficult problem in bias research is understanding whether differences in outcomes measured between protected groups are due to algorithmic/data bias or simply due to natural demographic variation [62]. In some contexts, there may be actual differences between groups, which the system should take into consideration. For example, in crime recidivism, it is known that men are more likely to commit future violent crimes than women with the same criminal history. Thus, there may be good reasons for employing gender-specific recidivism models to avoid women having their recidivism risk systematically overestimated, have started using such gender-specific tools [63].

Section 3.3 will describe different types of fairness-related biases that can occur in a machine learning context.

2.6 Bias Mitigation and Fairness-enhancing Methods

In this section, we will discuss techniques to enhance the fairness of machine learning methods. Many methods have been proposed since Pedreshi et al. published the first study on fair classification back in 2008 [46]. Chiefly, we can categorize them in four families: (i) pre-processing techniques, (ii) algorithmic modification techniques / in-processing, (iii) postprocessing techniques, and (iv) causal frameworks. The latter methods focus on detecting and removing discrimination by leveraging causal inference techniques. Causal approaches have become an active, and in our opinion, promising area of research. However, causal frameworks usually require access to graphs specifying causal relationships between different features, which can be difficult to obtain in practice. We will therefore concentrate on the three first mitigation strategies in this thesis.

Broadly, all bias mitigation related studies first specify some fairness measures that they wish to control, and then they propose techniques to control for said measures. Figure 2.10 shows an overview of the three main bias mitigation approaches, and then the next three sub-sections will discuss each separately.

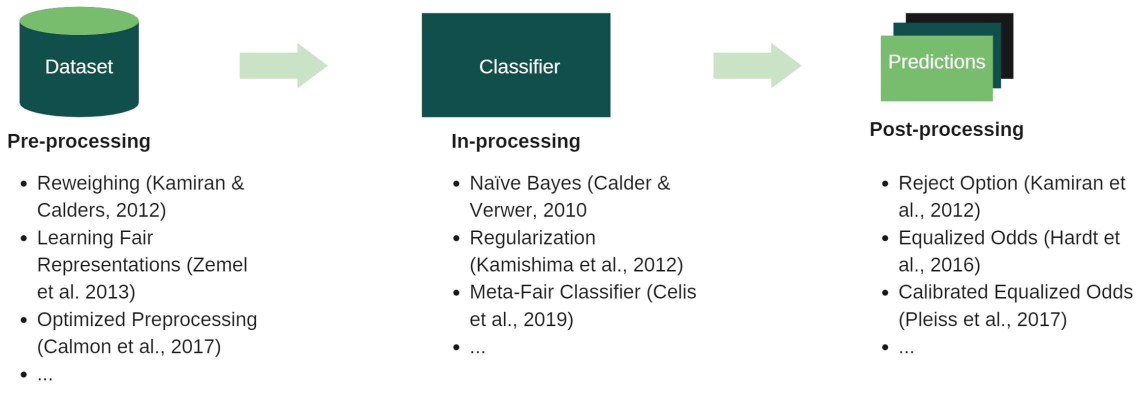

Dataset
Classifier
Predictions
Pre-processing
Reweighing (Kamiran & Calders, 2012)
Learning Fair Representations (Zemel et al. 2013)
Optimized Preprocessing (Calmon et al., 2017)
...
In-processing
Naïve Bayes (Calder & Verwer, 2010
Regularization (Kamishima et al., 2012)
Meta-Fair Classifier (Celis et al., 2019)
...
Post-processing
Reject Option (Kamiran et al., 2012)
Equalized Odds (Hardt et al., 2016)
Calibrated Equalized Odds (Pleiss et al., 2017)
...

Figure 2.10: *Approaches to Bias Mitigation*

(Adapted from Haas 2019)

2.6.1 Pre-processing

A machine learning algorithm is "only as good as the data it works with" [64], meaning it can only learn patterns present in the data. If that data contains bias, the algorithm will likely propagate the bias. Hence, it makes sense to attempt to change the characteristics of the input data in such a way that classification algorithms trained on this data achieve fairness with respect to their predictions. This is precisely the idea behind pre-processing techniques. An example is to remove existing correlations with the sensitive attribute, such that it becomes impossible to predict the sensitive attribute from the non-sensitive attributes. If we can do this while ensuring that the resulting distribution is as close as possible to the original data distribution, we may achieve decent accuracy while having guaranteed that any classifier trained on this data will be disparate impact-free. This is in essence the approach that Feldman et al. proposes [29]. Other pre-processing strategies include obfuscating the sensitive attributes and learning fair representations of the original data set [65][66], or changing the labels of some objects in the dataset in order to remove discrimination from the input data [67].

The main strength of the pre-processing approach is that the modified dataset can be used to train any algorithm afterwards.

We wish to mention briefly three specific approaches that we will make use of in our experiments:

Reweighing

The Reweighing method by Kamiran and Calders [67] attempts to de-bias the dataset by generating different weights for the training examples in each group-label combination.

Optimized Pre-processing

The Optimized Pre-processing method by [66] Calmon et al. learns a probabilistic transformation that edits the features and labels in the dataset with group fairness, individual distortion, and data fidelity constraints and objectives [43].

Disparate Impact Remover

The Disparate Impact Remover by Feldman et al. [29] edits feature values to increase group fairness while preserving rank-ordering within groups.

2.6.2 Algorithmic modification techniques / "in-processing"

This strategy consists of integrating the fairness constraints directly into a learning algorithm, by modifying the training procedure of the classifier, to ensure that the outputted model is fair. Numerous approaches exist. Zhang et al. apply adversarial learning to create unbiased models [68], Calders and Verwer propose embedding a non-discriminatory constraint into a decision tree classifier by changing its splitting criterion and pruning strategy [48], Kamishima proposes a regularization approach [69], while Celis et al. present a meta-algorithm with provable guarantees that can handle a number of fairness constraints and achieve near-optimal fairness in several datasets [70].

2.6.3 Post-processing

Post-processing techniques modify the outcome of an already trained model in an attempt to ensure fairness. This approach usually involves learning different decision thresholds for a given score function in order to remove disparate mistreatment or impact, or variants of re-labeling classifier predictions for individual instances in such a way that fairness is enhanced, like Kamiran et al. Pleiss et al., and Hardt et al. represent examples of [1], [71], [72].

A downside of most post-processing techniques is that they require access to the sensitive attribute at the decision time, an option that is frequently not available, due to disparate treatment laws.

Chapter 3

Related Work

Section 3.1 review some of the work done on fairness-related bias and discrimination. We review key papers, and then in chapter 3.2, present a taxonomy of bias that we have developed based on the literature review in section 3.1. Collectively, chapters 3.1 and 3.2 answer research question 1 (see section 1.3). Section 3.3 reviews the related literature on the topic of trade-offs between fairness and accuracy, and in the process, will answer research question 2 (see section 1.3).

3.1 A Review of Fairness-Related Bias in Machine Learning Literature

Our research question 1 (see section 1.3) asked what biases in algorithms and datasets cause unfair outcomes in machine learning systems. This section, along with section 3.2, will answer that question.

Much research has investigated the concept of bias in machine learning to understand, detect, and mitigate unfairness in algorithmic systems. However, fair ML literature is currently somewhat chaotic. Terminology is not aligned – sometimes even conflicting – and many terms and concepts lack clear or agreed-upon definitions. Historically, ML literature contains novel names for concepts that are already known under different names in statistics or other fields. For example, a feature, attribute, variable, covariate, predictor, and input all refer to the same thing. This tradition of coming up with new names for concepts instead of using established ones continues in fair ML literature, and occasionally those new terms conflict with other uses of the same term. As an example, Suresh and Guttag use *"representation bias"* to describe when "certain parts of the input space are underrepresented” [73], a concept that is already known as *undercoverage bias* in statistics. Another paper uses the same term (representation bias) to mean bias in the machine learning representation, where representation in this context may be a bag-of-words, or a deep recurrent neural network [74]. Most papers, though, simply talk about bias in very general terms like “training data bias”, or mention only context-specific bias applicable to limited domains, like image dataset [75], or social data [76]. This makes fair ML literature somewhat confusing to read, and makes it harder for practitioners to get a good grasp on the problem of unintended bias in ML.

As pointed out by Suresh and Guttag, many bias discussions in research papers are either too broad to be useful, or lack the shared terminology necessary to be understandable outside of specialty fields [73]. They began an important process of creating a comprehensive taxonomy that ML system developers can use in their work. We continue that process in this thesis (see section 3.2), hoping to bring greater clarity to the topic of bias in algorithmic decision systems. A handful of papers have presented collections and taxonomies of bias types, or mentioned examples of common bias types. Our thesis sums these up in a more comprehensive way than has been done before, and unify them under more commonly used names than what is sometimes used in various fair ML papers.

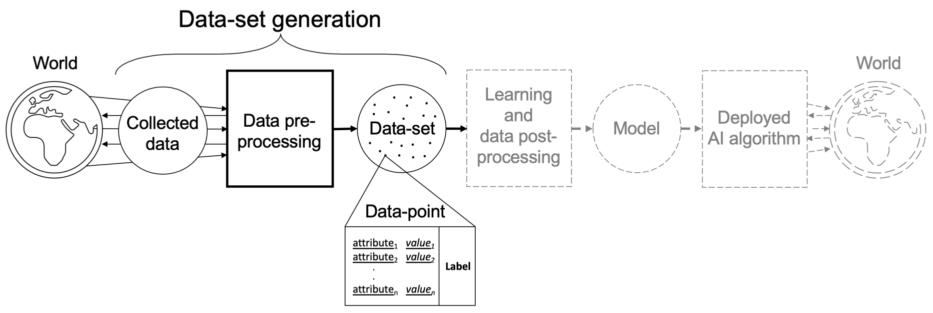


Figure 3.1: *An overview of the ML system development cycle.*

Friedman and Nissenbaum created the earliest taxonomy of bias in computer systems that we are aware of, sorting bias under three top-level categories: pre-existing bias, technical bias, and emergent bias [16]. A 2018 position paper by Dobbe et al. [77] expands on their work, with a particular focus on technical and emergent bias, positing that emergent bias can be understood by studying feedback mechanisms between algorithms and the environment upon which they act. We regard both these works as a foundation on which to build our understanding. Still, they do not go into enough detail that machine learning professionals can look to them for practical guidance for how to avoid biases in ML systems.

Baeza-Yates presents a taxonomy of 6 bias types that are specific to internet/web data [78], and Olteanu et al. present a different but related taxonomy, intended for understanding bias as it relates to *social data*, which include data coming from social media platforms like Facebook, as well as user-driven websites like Quora or Wikipedia [76]. Both works are lacking in generality, due to their focus on internet/web data and social data, respectively, but we draw elements from both works in this paper.

We also take into consideration work by Toralba and Efros presenting some bias types specific to image datasets [75], a paper by Rajkomar presenting biases in ML in the context of medicine [25], and a paper by Gupta et al. which also has a useful listing of some relevant bias types [79].

Suresh and Guttag describe five sources of bias in ML, while also explaining how these biases can lead to societal harms [73]. Their work subsumes the taxonomies and related results of Danks and London [24], Silva and Kenney [80], Calders and Žliobaitė [81], and Friedler et al. [82].

Finally, a 2019 survey by Mehrabi et al. [83] lists all bias types mentioned in [73], [76], [78], plus some from [84], along with short explanations of each. They further argue that the attempts in previous works to group bias types together is hard, due to feedback mechanisms, and attempt to model a categorization of bias definitions that take feedback loops into account.

In the next section, we will tie the available literature together and propose our approach to bias.

3.2 Our Approach to Bias

This section presents our approach to bias, which we have developed based on the literature survey in the previous section. More specifically, it lays out where in the ML system development cycle that biases can enter the process, what the reasons are, and what effects such biases can cause. We have never seen a presentation of this scope in the literature previously.

Table 3.1 gives examples of some simple bias issues. We refer to them as "simple" because, unlike in these examples, there are often not just one, but several intertwined sources of bias involved when the outcome of an algorithm is unfair.

Bias type	Example	Reason
Under-coverage bias	Darker-skinned females are misclassified by facial analysis algorithms with error rates up to 34.7%, while the maximum error rate for lighter-skinned males is 0.8% [85].	Under-representation of dark-skinned females in datasets.
Measurement bias	Same type of example as above (darker-skinned individuals are often misclassified by facial regcognition software), but different reason	Color balance settings and the dynamic range of cameras make it harder to capture high quality photos of people with dark skin than of people with light skin [86][61].
Label bias	The COMPAS crime recidivism tool has been shown to be biased against blacks in certain respects [3].	Crime recidivism prediction tools use future *arrests* as a proxy for future *crime*.

Historical bias / stale data	A Google Images search for "C.E.O." produced 11 percent women at a time when 27 percent of United States chief executives were women [5].	Historically, a lot less than 27% of CEOs were women, so if the search algorithm operated on historical data, it did not reflect current reality.
Aggregation bias	The COMPAS recidivism prediction tool has been shown to misclassify women as higher risk for violent crimes than they really are [63].	There are true differences between the genders in the sense that men are more likely to commit future violent crimes than women with the same criminal history [63], and the tool does not consider that.

Table 3.1: Some simple bias examples

3.2.1 Biases in the Model

Data preparation bias. As part of the machine learning pipeline, we need to decide what sets of data features to collect and use, and how to prepare them for use. A paper by Chouldechova and G'Sell shows that the choice of what features to use in the model can lead to subgroup differences in fairness [87].

Aggregation bias. Aggregation bias can occur when the underlying groups that the algorithm makes predictions on are in fact different, but the same model is used for all of them. In that way, the model's predictions may be inaccurate for all groups. Or if there is a majority group in the training data, the model will fit to it, and be inaccurate for other groups. In either case, it will yield unfair outcomes between groups [73].

As an example, in crime recidivism, it is known that men are more likely to commit future violent crimes than women with the same criminal history [63]. Thus, if we train a model on a sample consisting of men and women, the model will yield higher false positives for women than for men, even if they are equally represented in the sample. Similarly, in medicine, it is believed that levels of HbA1c, which doctors often use to diagnose diabetes, differ across races and genders [73].

Teams that develop predictive algorithms should work closely with domain experts to identify issues like this. And like we already mentioned in chapter 2.5, there may be good reasons for employing gender- or race-specific models, in order to avoid problems related to aggregation bias.

3.2.2 Biases in Data Collection or in the Training Data

Historical bias / stale data / temporal bias. If sampling or label distribution has changed over time, there may be a bias present in (old) data simply because times have changed [73], [79]. Example: In many countries, there are now more women than men going to medical school [88][89]. An old image dataset might contain mostly pictures of male doctors, making algorithms trained on it biased.

Label bias. Label bias can occur when we have unobservable / partially observable labels, such that one instead has to rely on proxies or labels decided by humans. We may regard it as a measurement error in the output y. We may also characterize it as the bias that occurs when the observed binary class labels on the training and testing set are influenced by sensitive attributes [90]. Label bias may be the hardest obstacle to overcome for fair machine learning [63]. It can occur in the following situations:

1) The training label is a proxy for the true label/output

In crime recidivism, the true label "did / did not commit a crime" is unknown, so crime recidivism algorithms have to rely on proxies like arrest records or convictions. In cities like Oakland, US, the police have historically concentrated more of their efforts in black neighborhoods than in white neighborhoods. Police tend to arrest more people in the areas they patrol than the areas they do not patrol, so in Oakland, drug-related arrest rates are 200 times higher for blacks than for whites, even though actual drug crimes are more evenly distributed between the groups [91]. Thus, an algorithm trained on that data will become biased against non-whites [77].

2) The training label is the actual outcome of a historical case

If the human who made the decisions contained in the training data was prejudiced or biased against some protected group, there may be a systematic bias in the training data, such that the algorithm is trained to become biased against this group too [77]. An example would be if a bank has been systematically and unfairly rejecting loans to people who should have been approved [69], or if an employer systematically has been hiring men instead of equally qualified female applicants.

3) Incorrect labeling due to prominence labeling bias or human ignorance

A labeler is more likely to apply an incorrect label to classes he or she is less familiar with or has less knowledge about. This typically affects less prominent classes. An example would be to label an image of a porpoise as a dolphin [79]. This points to the importance of involving domain experts in the data preparation process, especially in high-stakes domains.

4) Incorrect or inaccurate labeling due to cognitive or social biases, or social pressure

Biased labeling can occur if the labeler feels social pressure to label a certain way, i.e., when reporting on people's weight, or when labeling whether or not something is offensive [79]. A similar issue occurs when someone who wants to rate an item a certain way changes his mind because many others have rated it differently (social bias) [83], or because an authority rated it differently (authority bias). An example of a cognitive bias leading to biased labeling is to let irrelevant priors affect labeling, for example, labeling someone as a good person because they are handsome (the halo effect) [79].

Under-coverage bias. When a protected group is not sufficiently represented in the data for the algorithm to learn the correct statistical patterns, this is called *under-coverage bias*. Some sources also call it *under-representation bias* [92], or *minority bias* [25]. When the under-coverage is specifically in the training data, Suresh and Guttag calls it *"representation bias"* [73], and then they call it *"evaluation bias"* when the under-coverage is in the evaluation/test/benchmark data [73]. However, we feel that this is two sides of the same coin, and group it all under the umbrella of under-coverage.

There are several ways that under-coverage might occur. If the data points that the algorithm is trained, evaluated, or tested on is not randomly selected, this is called *sample selection bias,* or sometimes just *selection bias.* An example of this is if you have credit data from a bank that has been declining credit to a minority group without assessing their applications. Then that minority group will be under-represented in the sample [93]. A sub-category of sample selection bias is called *selective labels bias*. The prototypical example comes from the domain of judicial bail decisions. An algorithm is to be trained to make decisions about whether or not an arrested individual is likely to show up for court if he is released on bail. However, the only samples we can train the algorithm on are those individuals that a human judge historically decided to release. Those who were put in jail never got the chance to show up for court or not, so they are not included in our training data [94]. *Self-selection bias* occurs if your sample selection is based on people choosing whether or not to be a part of the sample [83], for example, by answering or not answering a survey you sent out, or by calling in after having heard a radio show. The latter case is also referred to as *voluntary response bias*, and not answering a survey is sometimes called *nonresponse bias*.

Even when individuals *are* sampled randomly and uniformly from the population, we, by definition, have fewer data points about minorities, leading to what is known as *sample size disparity*. Machine learning becomes more accurate when there is more data, so the algorithm may then work less well for members of the minority group [61].

In order to avoid under-coverage bias, developers should cooperate with domain experts to assess the fit between the collected data and the underlying population to be modeled [17].

Measurement bias. Once you have a sample of subjects, you need to measure the features/attributes/variables you are interested in. Measurement bias occurs when information collected for use as a feature/variable is inaccurate. This can lead to inaccurate outcomes. However, in our context, we are interested in unfair/biased outcomes, not inaccurate outcomes per se. If there is a greater proportion of measurement error in the features of one protected group than for another group, this can lead to unfair outcomes across those groups [50].

To illustrate the phenomenon, we use an example laid out by Mitchell et al.: *"accessing drug treatment (V) is used to predict child maltreatment (Y). If V is measured as accessing public treatment, its measurement will differ between poorer families and wealthier families, who may instead access private (i.e. unobserved) treatment"* [50].

Another example involves situations where a feature is an image. Images can not represent the world fully, they are limited by the technology in the camera, and limitations in this technology make cameras less adept at capturing details in dark-skinned individuals than in light-skinned individuals. This represents a form of measurement bias [61].

A special case of measurement bias is what Rajkomar et al. refers to as i*nformativeness bias:* "Features may be less informative to render a prediction in a protected group. Example: identifying melanoma from an image of a patient with dark skin may be more difficult." [25]

3.2.3 Biases in Interaction with Humans Who Use the System

Automation bias. When human decision-makers do not understand what potential bias may be present in the algorithmic decision-making system, they run the risk of trusting the system too much, inappropriately acting on inaccurate predictions [25]. This is a similar phenomenon as *decision-automation bias / technological halo effect*, which is the tendency of users of automated decision-support systems to become *"hampered in their critical judgment as a result of their faith in the perceived objectivity, neutrality, certainty, or superiority of the AI system"* [17].

Feedback loops. Feedback loops occur when decisions from the algorithm affect the data that is collected for future iterations of the training process [18]. The most common example involves predictive policing: If the system makes a biased recommendation that the police direct their resources towards a particular area, and the human decision-maker follow the recommendation even if it is incorrect to do so, the police will subsequently make more arrests in that area, simply because they are more likely to make many arrests in heavily policed areas. These new arrests are fed back into the system the next time it is trained, reinforcing its belief in the first prediction [91].

3.2.4 Biases in Interaction With the Subjects of the Algorithmic Decisions

Agency bias. Proving that an algorithm treats some group unfairly is hard, and certain minority groups may lack the education and resources to detect biases against them, or lack the influence to enforce corrections if biases are detected [25]. In that way, the bias is allowed to continue.

3.3 Methods for Evaluating Trade-offs Between Fairness and Accuracy

For at least a decade, it has been known that fairness-enhancing methods for ML classification are subject to a fairness/performance trade-off. To our knowledge, Kamiran, Calders, and Pechenizkiy [28] were the first to describe how achieving fairness comes at the cost of lowering the prediction accuracy for certain subgroups, and their results have been further cemented by later works [29]–[34], [95].

Zliobaite [96] studies the problem of making a binary classifier as accurate as possible under fairness constraints, in particular demographic parity. The author suggests that we should measure discrimination as the difference in classifier acceptance rates between the non-protected group and the protected group. He shows that the acceptance rate is directly dependent on the choice of the threshold value for the classifier. Changing acceptance rates leads to changes in baseline accuracy and baseline discrimination. Therefore, he argues that when evaluating the fairness of classifiers, one should take into account acceptance rates. Another interesting idea that the investigation in this paper suggests is that it might be beneficial to use the sensitive attributes during the training process, to quantify which portion of observed inequalities are justifiable, and which should be eliminated. Even though Zliobaite provides some methodological recommendations for comparison of fair classifiers and brief empirical analysis of trade-offs between accuracy and demographic parity, he does not attempt to generalize results beyond demographic parity, and does not quantify the relation between accuracy and fairness.

To our knowledge, Corbett-Davies et al. were the first to theoretically quantify the trade-offs between fairness and classifier performance, in an influential 2018 paper [31]. Their research investigates the cost of fairness from the point of view of trade-offs between fairness and public safety in the domain of bail decisions. The particular classifier they study is the COMPAS algorithm used in many US states to help judges decide whether defendants should be detained or released while awaiting trial. Even though COMPAS does not explicitly use race as an input, it has been shown to be racially biased, by systematically classifying black defendants as higher risk than they are [3]. The authors reformulate algorithmic fairness as a constrained optimization problem, where the objective is to maximize public safety while satisfying the following formal fairness constraints: statistical parity, conditional statistical parity, and predictive equality. Public safety in this context relates to the risk that a released defendant commits a violent crime. It is particularly the last of those three fairness metrics that COMPAS has been criticized for violating: the algorithm outputs higher false positives for blacks than for whites. In the paper the authors show that for the fairness metrics mentioned above, optimal classification requires applying separate thresholds for each protected group. These results suggest that an optimal trade-off between fairness and discrimination requires training a classifier with the objective of maximizing accuracy first, then set the separate decision thresholds in a post-processing step. However, in order to set different thresholds for different protected groups in a post-processing step, you need access to the sensitive attributes at decision time, which is often impossible due to disparate treatment laws or privacy concerns. Therefore, we argue that while interesting, the Corbett-Davies approach is probably not the best route forward in practical applications.

Menon and Williamson [34] derive similar results as Corbett-Davies et al. for traditional classification accuracy[1], and show that constraint-based mechanisms that do not use the protected attribute will have lower accuracy than one achieved by setting different thresholds. More specifically, they study the problem of learning with a fairness constraint independent of the choice of algorithm, aiming to find what is the best

1 Corbett-Davies et al. measure classifier performance by an objective they refer to as «immediate utility», formulated as a combination of the utility of a classifier and the cost of detaining individuals, but Corbett-Davies et al. states that this objective can be swapped with classification accuracy with the conclusions still standing.

accuracy that can be expected for a given level of fairness, and what is the nature of an optimal fairness-aware classifier? They use Disparate Impact factor and Mean Difference score as the fairness metrics for their analysis. Their results indicate that for cost-sensitive approximate fairness measures, the Bayes-optimal classifier is an instance-dependent thresholding of the class-probability function. Finding the Bayes-optimal fairness-aware classifier using these methods requires access to a theoretical population distribution, which in real-world settings might not be possible to achieve. And even though the paper provide a practical means of approximating the Bayes-optimal classifier, we still run into another problem: just as with the Corbett-Davies et al. paper, the Menon and Williamson results too relies on access to the sensitive attribute at decision time, which as we have described earlier may put us in conflict with disparate treatment laws. This places this research in the same category as the Corbett-Davies et al. paper when it comes to practical applicability. They derive several interesting theoretical results, for instance, in showing that the trade-off between fairness and accuracy depends on the similarity between the target and sensitive features, and in quantifying the degradation in performance by a measure of this alignment. However, just as with the previously mentioned papers, the major relevance to our research is in establishing that there is indeed a trade-off between accuracy and fairness. We shall not be making use of their results in our experiments.

Wick et al. [90] is a more recent work, and they consider two types of bias that lead to unfairness in machine learning: label bias and selection bias. To evaluate fairness, they look at the group fairness metric demographic parity, but the authors explain that their findings should be transferable to other notions of fairness. Their main finding is that when controlling for label and selection bias, the conflict between fairness and accuracy often disappears. This should not be surprising to anyone, as it is well known that the unfairness in ML models is often caused by data bias, as opposed to bias in the algorithm itself. Another interesting, and problematic, issue they raise is that many popular fairness assessments, such as equal odds and equal opportunity, involve error rates as measured against labeled data. As if the labeled data represents the ground truth, which of course, it might not do at all. The authors show that when measured against unbiased data, improving the fairness of classifiers may even increase accuracy as well, which makes sense if the protected group and non-protected group are, in fact, equal. If they are not, then the classifier should not be expected to produce equal outcomes for the groups. (Unless we want the classifier to create affirmative action, but that is another discussion entirely; a political one). The authors address the problem of not having access to perfect ground truth datasets, but mention that this problem can be partially overcome by data simulation. The downside of that method is that the artificially constructed dataset cannot quality controlled in a reliable way. Overall, the Wick et al. results add a piece to the AI fairness puzzle, by confirming what most practitioners already thought they knew, but there are no surprising findings here.

The reviewed papers, especially the Menon and Williamson [34] and Corbett-Davies et al. [31] papers, allow us to conclude that optimal classifiers under fairness constraints require separate thresholds for separate groups, and that applying such thresholds reduces classification accuracy. Wick et al. [90] show that this reduction does not have to occur when there is no underlying differences between the protected group and non-protected group with regards to ground truth outcome. In such cases, increasing the fairness of a classifier may also increase its accuracy.

All the papers reviewed have discussed methods for removing unfairness either while using static datasets, or when the discrimination comes from label bias or selection bias. It is well known, however, that unfair outcomes may result also from other types of biases, for instance measurement bias, other forms of under-coverage bias, and so forth. (See chapter 3.2 for a comprehensive overview of bias types). The best course of action in many cases is often to gather more or better data, which in theory might make the classifier fair without modifications, and hence without sacrificing accuracy. Sadly, this is often impossible in practice, which means that even in cases where the bias stems from non-representative data, or other sources of bias outside the algorithm, we still need methods for evaluating the fairness/accuracy trade-offs of different fairness-enhancing approaches.

This leads us to the next paper in our review, “The Price of Fairness – A Framework to Explore Trade-Offs in Algorithmic Fairness” by Christian Haas [35]. This paper is critical to our work, as our experiments build on Haas' suggestions for future research. Haas sets out to explore the relationship between fairness and performance in classifiers as a multi-objective optimization problem, by comparing Pareto fronts from different algorithms and bias mitigation approaches against each other. As we have seen in the other papers we have reviewed, fairness metrics may conflict with accuracy in the sense that increasing the fairness for a subgroup often leads to a decrease in overall accuracy for the classifier. Haas fills a gap in previous research by studying how to find a good balance between accuracy and the desired fairness metric. He proposes a generalized framework that can be adapted to different objectives, algorithms, and datasets. The framework consists of five separate stages and is depicted in figure 4.2. The first three stages involve selecting what dataset to work with, decide what the protected attribute(s) in the dataset will be (e.g. race, gender), define what metrics and objectives to consider, and select what classifier to use, along with any pre-processing, in-processing, or post-processing schemes to include. The final two steps involve calculating the Pareto front for each classifier and fairness approach, and then use the calculated trade-offs to determine the best level of fairness for the given classifier and approach.

The Haas framework uses Pareto-dominance sorting through the NSGA-II algorithm. By comparing multiple Pareto fronts generated by using different algorithms and fairness techniques, Haas is able to systematically analyze trade-offs between fairness and accuracy objectives. He demonstrates this in a case study, in which he uses classifiers with and without fairness enhancing schemes on the German credit dataset [97], with age as the sensitive attribute. SVM without fairness-enhancing techniques functions as the baseline. The classifier is trained with the GA approach, creating a Pareto front where the objectives are AUC and the fairness metric statistical parity, respectively. Against this Pareto front, he compares Pareto fronts made by training SVM after pre-processing by Reweighing, and by first training the SVM and then post-processing with the Reject Option Classifier (ROC). The fourth and final Pareto front is made by using an alternative classifier, the Meta-fair classifier, which is an in-processing technique to increase fairness. He also runs the same experiment one more time, changing the fairness metric from AUC to the individual fairness metric Theil index. By visually inspecting the resulting Pareto fronts, and calculating the Pareto fronts' hypervolumes, he can compare strategies analytically. In his paper, he skipped step five of his framework, but his results from step four makes it clear how one could proceed to select "best" trade-offs on the Pareto frontier. For example, the case study results show that optimization for the group fairness metric statistical parity resulted in observable differences between the approaches, with the SVM with ROC post-processing being the "best" solution, while the Meta-Fair approach is dominated by the other approaches (thus being the "worst"). Even when optimizing for the individual fairness metric, the ROC Pareto front yields the best AUC values in combination with the largest hypervolume. This implies that the Haas framework can be a useful tool for decision-makers and developers when making choices about what classifiers and fairness-enhancing methods to apply for their specific predictive analysis setting.

In research question 2 (see section 1.3), we asked what are the state-of-the-art frameworks or methods for investigating trade-offs between fairness and accuracy in machine learning classification systems. Our conclusion after having reviewed the literature is that there currently exist no established, agreed-upon system for this. Hence, one cannot say that there is a state-of-the-art. However, as our previous discussion shows, the Haas framework shows promise, and we will design experiments with the intent of confirming or disproving its usefulness (chapter 5).

Chapter 4

Approach

This chapter will first present the development tools and programming libraries used for this thesis. It will then describe the research method we will use to answer research question 3 from section 1.3, and finally, present our architecture.

4.1 Development Tools and Libraries

We do not wish to build the necessary tools for our experiments from scratch; therefore we will rely on publicly available libraries and platforms. This section describes the ones we have chosen to use.

4.1.1 scikit-learn

There are many machine learning tools and libraries to choose between; from remote tools like Google Prediction API, AWS Machine Learning, or Microsoft Azure Machine Learning, to local tools like GoLearn, TensorFlow, Keras, PyTorch, and scikit-learn. Many of these are unnecessarily complex for our needs, emphasizing deep learning algorithms, so we settled on scikit-learn, which is an easy to use, open-source machine learning library built on top of SciPy, NumPy, and matplotlib. It provides several well-established algorithms for supervised and unsupervised learning via a consistent interface in Python. It is licensed under a permissive and simplified BSD license, and it is well suited for classification and regression tasks. Importantly, for our purposes, it contains a good implementation of a support vector machine (SVM) classifier. The next section will go into greater detail on our set-up of the SVM for this thesis.

4.1.2 SVM

The general background theory on support vector machines was given in section 2.1.1, but to recap briefly, an SVM tries to achieve two goals simultaneously: a hyperplane with the largest minimum margin, and a hyperplane that correctly separates as many data points as possible. The C parameter regulates this trade-off: a low C gives a larger margin, but more misclassified data points, and a high C gives fewer misclassified instances but a narrower margin.

Like we already mentioned in section 2.1.1, for our experiments, we will use the Gaussian radial basis function (RBF) kernel, which has the following equation:

$$\phi_{\gamma}(\boldsymbol{x}, l) = \exp(-\mu \|\boldsymbol{x} - l\|^2), \mu > 0$$

where **x** = (x_1, x_2), *l* is the center from which the euclidian distance of **x** is measured, and γ is the kernel parameter.

The optimal value for C and γ depends on the dataset, and we will use the genetic algorithm NSGA-II, a multi-objective optimization algorithm, to optimize both. Section 4.3 lays out the details for how we go about doing this.

4.1.3 AI Fairness 360

AI Fairness 360 (AIF360) [43] is an open-source Python toolkit for detecting, understanding, and mitigating algorithmic unfairness and bias. It is developed by researchers at IBM, and is available under an Apache v2.0 license. It provides a common framework for fairness researchers to share and evaluate algorithms, and it also aims to facilitate the transition of fairness research algorithms to use in an industrial setting. It contains a comprehensive set of fairness metrics for datasets and models, as well as pre-processing, post-processing, and algorithmic techniques for mitigating and evaluating bias in datasets and models. A host of common datasets is also provided through the `StandardDataset` class, where datasets can be loaded in different manners simply by passing different arguments to the constructor, making it easy to configure the loading procedure at runtime.

4.1.4 NSGA-II

Like we have described in section 2.3, genetic algorithms are well suited for determining Pareto dominance relationships in multi-objective optimization. For our thesis, we will use an implementation of a genetic algorithm called NSGA-II (Non-dominated Sorting Genetic Algorithm), designed by Deb et al [98]. NSGA-II is famous for fast non-dominated search, and compared to other multi-objective evolutionary algorithms (MOEAs), like PAES and SPEA, it will often find a better spread of solutions and better convergence near the true Pareto-optimal front. It uses elitism, as well as the diversity-preserving mechanism called crowding distance, both of which we discussed in section 2.3. Figure 4.1 shows a flowchart for NSGA-II.

The steps of the NSGA-II algorithm goes like this:

Step 1: Create a random, initial parent population *P* of size *Z*

Step 2: Evaluate objective functions, and sort the random parent population based on non-domination

Step 3: For each non-dominated solution, assign a rank equal to its non-domination level (1 is the best level, 2 is the next best level, and so forth)

Step 4: Create an offspring population *Q* of size *Z* using binary tournament selection, crossover, and mutation operators on *P*.

Step 5: Evaluate objective functions of the offspring population

Step 6: Combine the two populations to form *R* of size 2*Z*

Step 7: Use the fast non-dominated sorting algorithm to sort according to non-domination, and rank the population according to its non-domination level

Step 8: Identify a Pareto front of best solutions, F_1

Step 9: If the size of F_1 is larger than *Z*, reduce it by crowding selection, store the solutions in F_1 in a new set *P**, then move to step (8)

Step 10: If the size of F_1 is smaller than *Z*, store the set of solutions in F_1 in a new set *P**, remove them from *R* and keep repeating the process of finding new Pareto fronts, storing the solutions in *P** and removing them from *R* until the combined number of solutions in $P*$ and the next Pareto front, F_k, exceeds Z. Then reduce F_k by crowding selection and store it in $P*$.

Step 11: The selected solutions in $P*$ now form a new population P, and the remaining solutions in R are rejected.

Step 12: Repeat from step (4) until some termination criterion is met

Step 13: Perform a final Pareto selection to get the best solutions from the final population

Implementation specifics for crossover and mutation operators, and the evaluation of objective functions, are given in section 4.3.

Arguably, the defining components of NSGA-II are fast non-dominated sorting, and the use of crowding distance and elitism. See section 2.3 for a description of both. Elitism makes sure that the best solutions from each generation is carried forward so that they do not have to be re-discovered, and crowding distance and fast non-dominated sorting is involved in determining what individuals get selected for reproduction. The fast non-dominated sorting algorithm creates ranked Pareto fronts, each Pareto front containing solutions that do not dominate each other, and the crowding distance algorithm is used to preserve diversity by applying tournament selection to reduce the population, where solutions compete by comparing their crowding distance. Selection is performed by binary tournament. In the binary tournament are considered first the rank, and then the crowding distance. Between two solutions with differing non-domination ranks, the solution with the lower rank is preferred. If both solutions in the tournament belong to the same Pareto front, then we prefer the solution that is located in the least crowded region of the frontier.

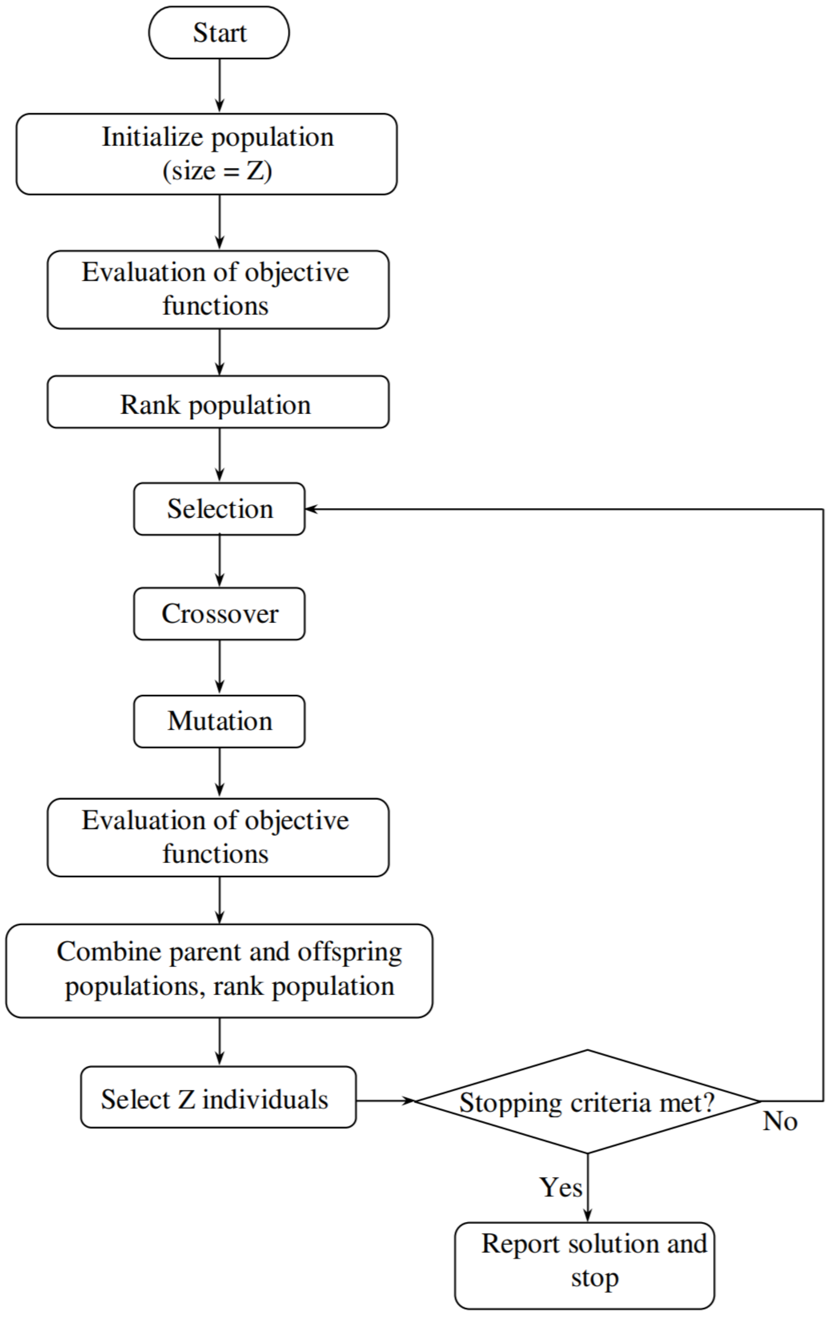

Start
Initialize population (size = Z)
Evaluation of objective functions
Rank population
Selection
Crossover
Mutation
Evaluation of objective functions
Combine parent and offspring populations, rank population
Select Z individuals
Stopping criteria met?
No
Yes
Report solution and stop

Figure 4.1: *Flowchart of NSGA-II*

4.2 Method Description

For our experiments, we will implement the Haas framework for exploring trade-offs in algorithmic fairness [35], that we gained some familiarity with, in section 3.3. The framework consists of the five stages depicted in figure 4.2. They are:

1. selection of dataset and protected attribute
2. defining metrics and objectives that should be considered
3. selection of classifiers and algorithms that will be used to calculate a Pareto front, and additionally, one might include in this step a selection of pre-processing, in-processing, or post-processing steps for increasing fairness
4. calculation of the Pareto front for each classifier and fairness approach
5. using the calculated trade-offs to determine the best level of fairness for a given algorithm and approach

The purpose of the framework is to use multi-objective optimization to try to find a good balance between performance metrics such as accuracy, and fairness metrics, for a given predictive analytics problem.

Steps 1 & 2 of the framework will be covered in the next chapter, where we lay out our experiments. For step 3, we have chosen to use a support vector machine. SVMs are powerful, versatile, and particularly well suited for classification of complex but medium-sized datasets, which the typical datasets used in ML fairness-research are.

For step 4, the multi-objective optimization that calculates the Pareto fronts, we will be using NSGA-II (see section 4.1.4), which is a fast genetic algorithm that has been shown to be highly effective for performing Pareto-dominance sorting [99]. We could have used a variety of other evolutionary algorithms instead, but decided on NSGA-II for two reasons: it is what Haas used, and we have good knowledge and first-hand experience with it from a previous NTNU course.

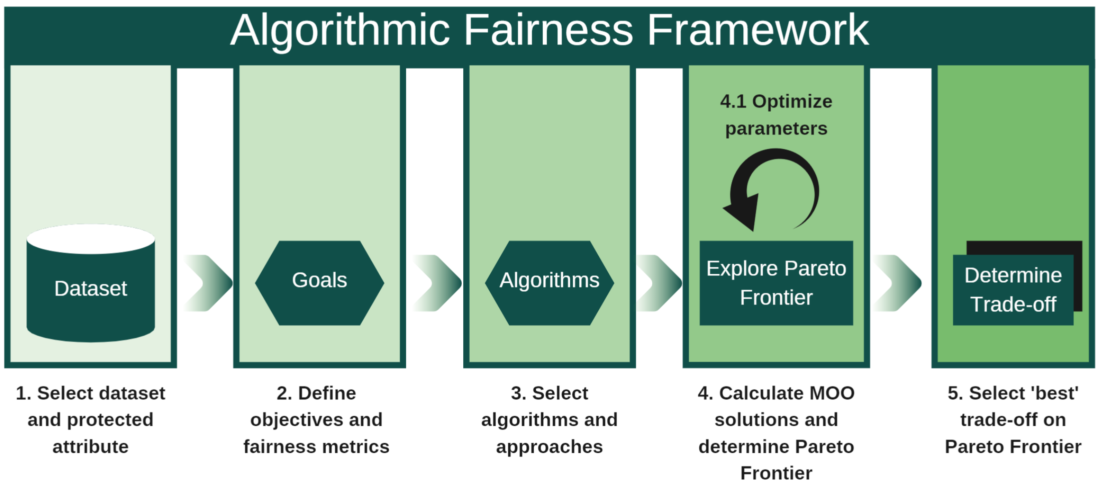

Algorithmic Fairness Framework
Dataset
Goals
Algorithms
4.1 Optimize parameters
Explore Pareto Frontier
Determine Trade-off
1. Select dataset and protected attribute
2. Define objectives and fairness metrics
3. Select algorithms and approaches
4. Calculate MOO solutions and determine Pareto Frontier
5. Select 'best' trade-off on Pareto Frontier

Figure 4.2: *Framework to explore Algorithmic Fairness Trade-offs*

(Adopted from Haas 2019)

In any machine learning task in general, and in any classification task in particular, two of the most important steps are feature selection and parameter tuning. Feature selection identifies what features to use from the original dataset, and that choice greatly impacts the speed with which the model can be trained. It may also affect how many examples are needed, because more features increase the complexity of the pattern recognition problem, increasing the demand for training examples to generalize from. Therefore, the goal of feature selection can be said to be to allow the model to generalize well with as few features as possible. Parameter tuning is necessary in order to make sure the classifier performs optimally under the constraints of the dataset it has been given. We discussed hyperparameters for support vector machines using the RBF kernel function in sections 2.1.1 and 4.1.2, and we identified them to be the penalty parameter C, and the kernel parameter γ. Huang and Wang have shown that genetic algorithms are effective for parameter tuning [100], and that they achieve better results than the most frequently used traditional parameter tuning method, Grid Search, while simultaneously being able to perform automated feature selection. For our experiments, we will be using the NSGA-II to perform parameter tuning and feature selection for the SVM, while optimizing two objectives: one performance objective, and one fairness objective.

Haas does not go into much architectural detail about how they implemented the framework in their case study experiments, so we do not know to what degree our implementation will match theirs. However, it is clear that to implement the framework, we have to decide how to design the NSGA-II chromosomes, how to design the crossover and mutation operators, and how to generate the initial population of solutions. The next section will lay out the details for these decisions.

4.3 Architecture

The task at hand is to optimize hyperparameters for the support vector machine and perform feature selection. Our genetic algorithm, NSGA-II will do both tasks, so the only architectural choices left are those relating to NSGA-II. Chromosome design, how to create the initial parent population of chromosomes, choosing the fitness function to help the algorithm determine which chromosomes/solutions are better than others, and finally, determining what type of crossover and mutation operators to use to create offspring. In evolutionary algorithms in general, one would also have to decide how to create diversity in the population, and how to select what chromosomes in the parent population to select for reproduction. Those decisions, however, are already baked into the NSGA-II algorithm. Crowding distance sorting (see section 2.3) takes care of diversity, and the NSGA-II selection mechanisms were described briefly in section 4.1.4.

The next sections provide detail about our chromosome representation, how to create the initial population, mutation and crossover operators, and evaluation of the objectives, respectively.

4.3.1 Chromosome Representation

When using genetic algorithms, one of the most important - and challenging - aspects is the design of the chromosome. The chromosome represents the solutions to the problem at hand. The task we wish to give our genetic algorithm is to perform hyperparameter tuning and feature selection while balancing our two objectives (accuracy and a fairness

measure, respectively). Therefore, each chromosome must represent a set of hyperparameters and features.
As we saw in section 2.3, the representation of a chromosome can be split into two parts: the genotype and the phenotype. The former is the encoding that is used by the GA, and the latter is the encoding that represents the real-world solution. In our architecture, the phenotype is the real-valued parameters for the support vector machine and the feature mask representing the features to use. There are several options for representing this phenotype as a genotype, and we have chosen a simple one: the binary encoding system. In it, a list of bits is all that is required. That allows for easy implementation, and it makes it easier to design genetic operators.
Some genes in each chromosome will represent the C, some genes will represent γ, and the rest will represent the feature mask. See figure 4.3 for a visualization of the representation. Each C_k represents a bit in C, such that the *n* in $C_1,...,C_n$ represents the number of bits of C. Similarly with γ, which is granted *m* bits, and the features, which are given *q* bits. Naturally, *q* must equal the number of features in the dataset.

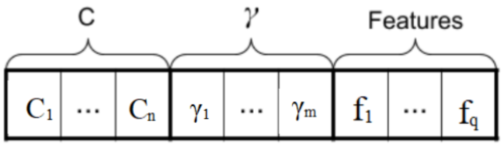


Figure 4.3: *Chromosome representation, consisting of three parts: C,* γ, *and the feature mask.*

For feature mask genotype decoding, we let a bit value of 1 mean that a feature is kept, and a bit value of 0 means that it is discarded.
We wish to let the bit representations of C and γ be decoded into floating-point numbers, to allow for higher precision with a shorter length chromosome. For this, we will use the IEEE 754 standard. Under the IEEE 754 standard, a floating-point number is specified by:

- a base *b,* which will be 2 in our case
- a precision *p*

- an exponent range from *emin* to *emax,* with *emin* = 1 - *emax*

The first bit in the IEEE 754-standard is reserved for the sign of the number, but for our purposes, the sign will always be positive. Thus, we can dispose of it. Next follows a number of bits reserved for the exponent, and the remaining bits represent the significand. The following formula gives the floating-point value *f* of a binary number, using the IEEE-standard:

$$f = 2^e \frac{s}{2^{p-1}}$$

where *s* is the significand, *p* is the number of digits in the significand, and *e* is the exponent. Figure 4.4 shows an example of a layout for 32-bit floating points.

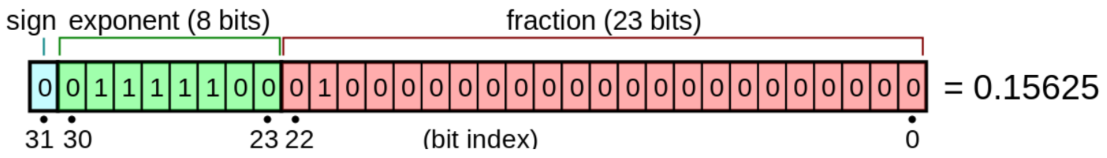


Figure 4.4: *32-bit floating point layout.*

To calculate the floating point value of our binary representation, the following equation is used:

$$(\sum_{n=0}^{p-1} 2^{-n}) * 2^e$$

4.3.2 Generation of the Initial Population

A common strategy for generating initial populations is to generate them randomly. That ensures diversity in the population, and it is the strategy we will use. We do that by randomly generating 1's or 0's for each position in each chromosome.

4.3.3 Crossover and mutation

We described crossover and mutation operations in detail in section 2.3. For binary representations, it is fairly straight forward to implement both, and for this thesis, we have chosen the following strategy:

Crossover operator: choose a random crossover point, and split each parent chromosome in two at that point. Then combine the parts in the opposite way, creating two offspring chromosomes. This operation is visualized in figure 2.5.

Mutation operator: choose a random bit in the chromosome, and flip the value of the bit. If it used to be a 0, it becomes a 1, and vice versa. This operation is visualized in figure 2.6.

4.3.4 Evaluation of objectives

One of the steps of NSGA-II is to rank the population (see figure 4.1). This is a step for NSGA-II has to do to be able to perform non-domination sorting. Before ranking, we need to evaluate the objectives for each chromosome in the parent and offspring populations. We need two values; one representing the performance measure of the classifier, and one representing a fairness measure. In order to obtain those values, we need to run the scikit-learn SVM classifier, as well as apply one of the mitigation approaches from the AIF360 toolkit. This must be done for each chromosome in the total population. The steps for evaluating the objective functions are as follows:

1. **Scale test set and training set**: both sets are scaled using the following function:

$$x_{scaled} = \frac{x - \overline{X}}{s}$$

   where $\overline{X}$ is the mean of the training samples, and s is the standard deviation of the training samples.

2. **Convert genotype to phenotype**: this is done in the way that is explained in section 4.3.1

3. **Keep feature subset**: The phenotype representing the feature mask is used to keep the chosen feature subset and discard the other features from the training and test sets.
4. **Train SVM classifier**: The phenotypes of the values of C and γ from the chromosome is used as parameters for the SVM classifier, which in turn is trained using the training set.
5. **Test SVM classifier**: The trained classifier is used to predict scores for test samples. These scores are assigned labels using a classification threshold of 0.5, where scores above the threshold is assigned the positive label, and scores below the threshold is assigned the negative label.
6. **Apply bias mitigation method**: Apply bias mitigation method from the AIF360 library. This step must be done earlier in the cycle if pre-processing is the chosen mitigation method.
7. **Calculate values for the objectives**: Calculate the desired accuracy and fairness measures, based on the predictions from step 5. Collectively, these are used to rank the chromosome.
8. **Return values**

The calculated values for each objective is then used when performing non-dominated sorting.

4.4 Implementation Details

All code is written in Python3, and resides in a Github repository, along with instructions for how to run the code (in the file README.md)[2]. Table 4.1 lists the Python modules we have used to run our code, and their version numbers.

2 https://github.com/gunnaja/ml_fairness_tradeoffs

Module	**Version**	**Remarks**
`numpy`	1.17.4	
`scikit-learn` (sklearn)	0.22	
`aif360`	0.2.3	
`pandas`	0.25.3	Our code does not run with newer versions of pandas. We don't know the reason for this.
`matplotlib`	3.1.2	

Table 4.1: List of Python modules required to run our code

The machine learning library **`scikit-learn`** that we chose to use for the SVM algorithm is described in section 4.1.1. For bias mitigation, we used the IBM **`aif360`** toolkit, which is described in section 4.1.3. Other viable choices would have been **`Themis-ML`** or **`Fairness Comparison`**, but wishing to stay close to the Haas implementation, we decided on using **`aif360`**. **`Numpy`** was used to represent and perform calculations on the chromosomes in our NSGA-II implementation. It is also a requirement for some of the other modules. **`Pandas`** was a requirement for the **`aif360`** module. Finally, **`Matplotlib`** was used to plot the results from our experiments.

Instructions on how to run the code can be found in the README.md file in the Github repository

For non-separable datasets, there is a risk that the SVM classifier, for some values of C and γ, will try to fit the training set almost indefinitely. To avoid this, we set the maximum number of iterations to 15 thousand.

Chapter 5

Experiments and Results

This chapter will present our experiments, built on the approach we laid out in chapter four. The purpose of the experiments is to answer research question 3. Section 5.1 explains our experimental plan, section 5.2 explains the experimental setup, and the final section, 5.3, presents our results and analysis.

5.1 Experimental Plan

In section 3.3 we found that the framework proposed by Haas in their 2019 study "The Price of Fairness - A Framework to Explore Trade-offs in Algorithmic Fairness" [35], for investigating trade-offs between accuracy and fairness in machine learning systems, is our best candidate for being called state-of-the-art. It has not yet stood the test of time, but there are no better candidates that we are aware of. We will, therefore, implement the Haas framework, and design experiments for it in order to answer research question 3 (see section 1.3). Our experiments are based on the suggestions Haas provides as potential avenues for future research in their paper. We will apply additional bias mitigation approaches, apply the framework on a different dataset from what they used, and we will also investigate trade-offs between accuracy and fairness for more fairness metrics than they did in the Haas paper.

In the Haas paper, they use the following mitigation approaches: MetaFair for in-processing, the Reject Option classifier (ROC) for post-processing, and the pre-processing they use the Reweighing algorithm. We will extend this by using two different pre-processing methods, namely Optimized Pre-Processing, and Disparate Impact Remover.

In the Haas study, they use the German Credit Dataset. We will extend their work by using a different dataset, namely the COMPAS dataset described in section 5.2.1.

5.2 Experimental Setup

This section describes the setup for the experiments we will be performing in this thesis. The first section describes the dataset. Section 5.2.2 describes the fairness and performance measures we will evaluate. Section 5.2.3 lays out our chosen parameter settings for the genetic algorithm. Section 5.2.4 gives some information about the GA chromosome. Section 5.2.5 presents our choice of algorithm and mitigation technique. Finally, section 5.2.6 summarizes the choices in a table, for ease of reference.

5.2.1 Dataset and Dataset Pre-processing

We experiment with a real-world dataset: the ProPublica COMPAS risk assessment dataset [101]. The dataset, gathered by ProPublica, contains information about all criminal defendants subject to screening by the COMPAS risk assessment tool in Broward County, Florida, in 2013-2014. It consists of data about more than 7 thousand pre-trial criminal defendants, and relates to recidivism risk prediction, i.e., predicting if a criminal defendant will commit an offense within a certain future time. The dataset contains a number of features such as the age and number of prior criminal offenses for the defendant, as well as a class label indicating whether or not the person recidivated within two years of their arrest. The former is the positive class, and the latter, the negative class.

We will be using the "Risk of Recidivism" label, and the sensitive attribute we will use is *race*, where White/Caucasian is the privileged group, and the others are combined into a "Non-Caucasian" unprivileged group.

The **`aif360`** (see section 4.1.3) tool provides a version of the COMPAS data set that is pre-processed in the same way as the original [43]. This includes removing rows with missing data, and selecting only the most relevant features.

The `aif360` tool uses one-hot-encoding on the charge description feature, adding another 389 features to the dataset. We will remove those features, and use the following ten: *sex, age, age category, race, juvenile felony count, juvenile misdemeanor count, juvenile other count, priors count, charge degree,* and *charge description*. *Age* and *charge degree* is one-hot-encoded, bringing the feature total up to 12. Examples with missing data is removed, which brings the total number of examples down to 6.172.

We split the dataset into an 80/20 split for training and testing. Details about the split can be seen in table 5.1.

Dataset type	**# of features**	**# of data points**	**% of total dataset**
Training set	12	4937	80%
Test set	12	1235	20%
Total dataset	12	6172	100%

Table 5.1: The table shows the two dataset splits with their respective amount of data points

Relevant to our further investigations, research by Dressel and Farid [102] have been unable to reach higher accuracies than 65-67% in their experiments on this dataset, using both linear and non-linear classifiers, leading them to conclude that the COMPAS dataset is neither linearly nor non-linearly separable.

5.2.2 Metrics

Performance metric

We choose to use plain binary accuracy as our performance metric. The metric is explained in section 2.4, but to recap, accuracy in binary classification settings is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

We are aware of the potential problems associated with this metric, as explained in section 2.4, but we had no such problems in our experiments.

We will investigate the accuracy-fairness trade-offs of our methods using two different fairness metrics, namely Statistical Parity Difference (SP_{Diff}) and Theil Index. To recap from section 2.4, Statistical Parity Difference is a group fairness metric defined as the difference in the probability of favorable outcomes between the unprivileged and privileged groups, and Theil index is an individual fairness metric.

5.2.3 NSGA-II Parameters

Four parameters need to be adjusted for the NSGA-II algorithm to run: number of generations, population size (number of chromosomes / candidate solutions), mutation rate, and the crossover probability. There are no hard-and-fast rules for how to go about selecting these parameters, but generally, because a longer chromosome exponentially increases the size of the solution space, longer chromosomes may require larger population sizes and higher number of generations. As a rule of thumb, we will attempt to use a population size *N* that is at least as big as the number of features, and a mutation rate of about 1/*N*. Table 5.2 shows our parameter settings.

Parameters	**Value**
Number of generations	100
Population size (number of candidate solutions / chromosomes)	100
Mutation rate	0.1
Crossover probability	0.8

Table 5.2: NSGA-II parameters for our experiments

5.2.4 Chromosome length

The chromosome representation is described in section 4.3.1. We will use a chromosome length of 42, where the first thirty bits consists of 15 bits each for the C and γ values, and the last twelve bits represent the 12 features we wish to use from the COMPAS dataset. The C value will be decoded into a range of $[2^{-16}, 2^{16}]$, while the γ value will be decoded into a range of $[2^{-10}, 2^{3}]$.

5.2.5 Algorithm & Mitigation Technique

As explained in section 4.1.2, we will be using the standard sklearn SVM classifier for our experiments, with a Gaussian radial basis function (RBF) kernel. We will run the classifier with four different mitigation techniques:

1. No mitigation, just the SVM classifier alone. This represents our baseline.
2. The Reweighing algorithm + SVM: The pre-processing bias mitigation technique called Reweighing will be performed on the training data before it is used to train the classifier. Reweighing is part of the **aif360** toolkit.
3. Optimized Pre-processing + SVM: The pre-processing bias mitigation technique called Optimized Pre-processing from the **aif360** toolkit is applied on the training data before it is used to train the classifier.
4. Disparate Impact Remover + SVM: The pre-processing bias mitigation technique called Disparate Impact Remover from the **aif360** toolkit is used on the training data before it is used to train the classifier.

For each of the approaches, the genetic algorithm NSGA-II is used to optimize the feature selection and classifier parameters. As mentioned, all three mitigation techniques are pre-processing methods. The methods are described in section 2.6.

5.2.6 Setup Summary

Table 5.3 summarizes the experimental setup.

Parameter	**Values**	**Description**
Dataset	The COMPAS dataset	See section 5.2.1
Protected / sensitive attribute	Race	Privileged: Caucasian Unprivileged: Non-caucasian
Algorithms	SVM alone Reweighing + SVM Disparate Impact Remover + SVM Optimized Pre-Processing + SVM	The algorithms along with their respective bias mitigation strategies.
Performance metric	Accuracy	See section 2.1
Fairness metrics	Statistical Parity Difference, Theil Index, Equal Opportunity Index, Disparate Impact, Average Odds Difference	See section 2.4
NSGA-II parameters	Generations: 100 Population size: 100 Mutation rate: 0.1 Crossover rate: 0.8	The parameters used for the genetic algorithm.

Table 5.3: Summary of experimental setup

5.3 Results and Analysis

We used four different algorithms for our experiments: a baseline, no-mitigation approach (an SVM classifier alone), and three SVM classifiers trained on data that was pre-processed by three different pre-processing mitigation methods. For a short description of the algorithms, see section 2.6, and for an in-depth description of the architecture used to perform the experiment, see section 4.3. The experiments were run for five different scenarios, one for each of the fairness metrics we wish to optimize for (see table 5.4).

Scenario name	**Accuracy metric vs. fairness metric**
Scenario 1	Binary Accuracy vs. Statistical Parity Difference
Scenario 2	Binary Accuracy vs. Theil Index
Scenario 3	Binary Accuracy vs. Equal Opportunity Difference
Scenario 4	Binary Accuracy vs. Disparate Impact
Scenario 5	Binary Accuracy vs. Average Odds Difference

Table 5.4: *The scenarios in our experiments*

In scenarios 1, 3, 4, and 5, we optimize for group fairness metrics: Statistical Parity, Equal Opportunity, Disparate Impact, and Average Odds Difference, respectively. In scenario 2, we optimize for the individual fairness metric Theil Index. See table 2.2 for an explanation of each of the metrics. While the **aif360** tool outputs the optimal score for all the fairness metrics as zero, we have chosen to reverse this in our plots, such that all figures display the score as [1 - fairness score], leading to 1 being the best possible fairness score obtainable. Likewise, 1 is the best score for accuracy as well. Hence the perfect score would be (1, 1), in the upper right corner of the plots.

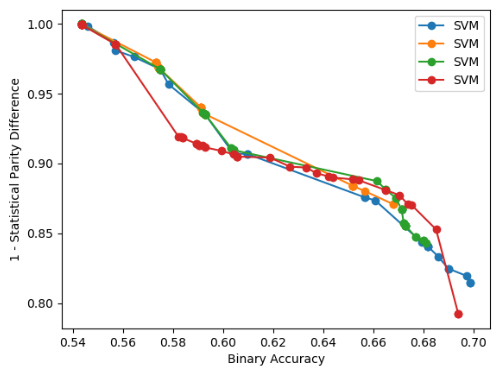
The performance of multi-objective optimizers are often compared using hypervolumes, also called S-metrics, or Lebesgue-measures. However, our results are easy to interpret by visual inspection of the plotted graphs, so we will limit ourselves to that.

Some runs of the algorithms produced clearly sub-optimal Pareto fronts, so we decided to discard those, and to run each algorithm enough times that we would be able to select four good Pareto fronts for each algorithm and each fairness metric to present in this paper.

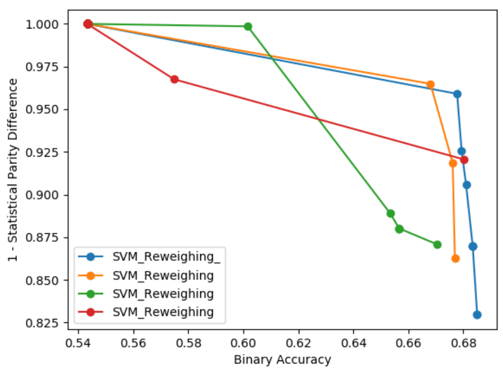
The reason we wanted four good Pareto fronts for each algorithm for each scenario, is twofold: (1) to show that the hyperparameter settings we described in chapter 5.2 produce sufficiently consistent results for us to be able to make conclusions based on our experiments, and (2) to illustrate an important point: that even when running the algorithms numerous times and selecting the "best" Pareto fronts, the results of applying the mitigation methods vary. In other words, the mitigation methods do not produce consistently good bias mitigation. Sometimes the results are good, but sometimes there is very little, or even no, fairness enhancement compared to the baseline no-mitigation approach.

Figure 5.1 shows the results for the four algorithms for scenario 1, with Accuracy along the x-axis, and Statistical Parity Difference along the y-axis. Figure 5.2 shows the results for the four algorithms for scenario 2; with Accuracy along the x-axis and Theil Index along the y-axis. Figure 5.3 shows the results for the four algorithms for scenario 3; with Accuracy along the x-axis and Equal Opportunity Difference along the y-axis. Figure 5.4 shows the results for the four algorithms for scenario 4; with Accuracy along the x-axis and Disparate Impact along the y-axis. In all four figures, the upper left plot, (a), shows the baseline SVM algorithm run alone on the COMPAS dataset, with no bias mitigation method. Plot (b) shows the results of four runs where the Reweighing bias mitigation method is applied on the data before the SVM classification. Plot (c) shows the results of four runs where Disparate Impact Remover is applied before the SVM, and finally, plot (d) shows the results of four runs where Optimized Pre-processing is applied before the SVM. Each point on the Pareto fronts represents one chromosome / solution, and because of the nature of Pareto fronts, they all represent non-dominated accuracy-fairness combinations. Although some of the plots may look like they are not, all the plotted Pareto fronts are strictly decreasing.

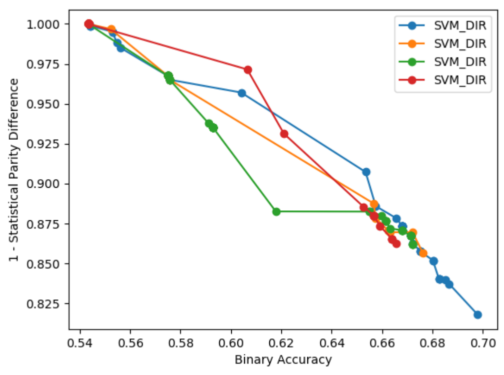
Immediately, it can be seen in figures 5.1-5.4, that for all five scenarios, the frontiers for the pure SVM algorithm follow each other rather closely, giving us the answer we hoped for, namely that the NSGA-II parameters we have chosen produce consistent results.



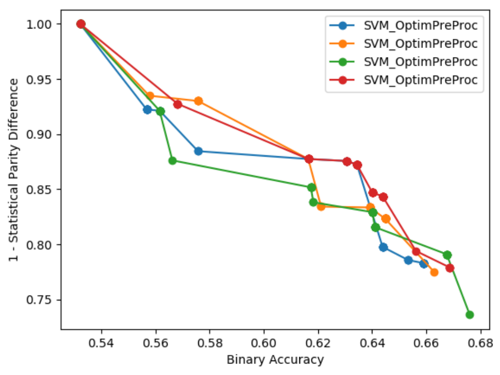
a) No mitigation approach, just baseline SVM



b) Reweighing + SVM

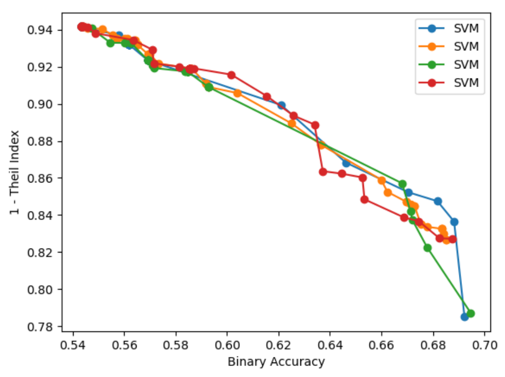


c) Disparate Impact Remover + SVM

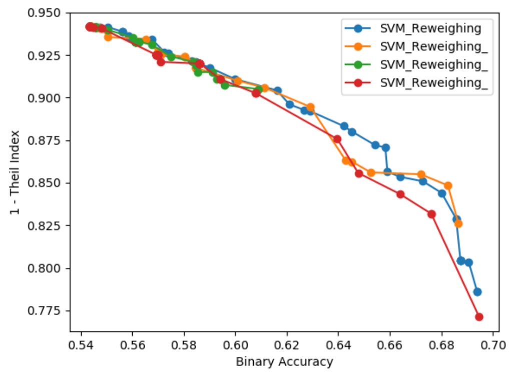


d) Optimized Pre-processing + SVM

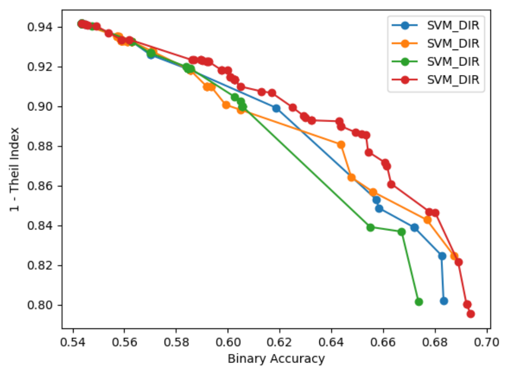
Figure 5.1: *Pareto fronts for Binary Accuracy vs. Statistical Parity Difference on the COMPAS dataset. Four runs for each bias mitigation approach.*



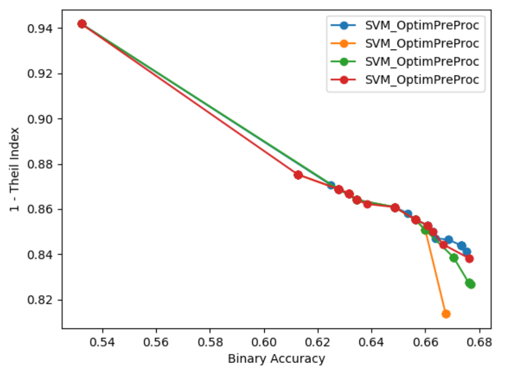
a) No mitigation approach, just baseline SVM



b) Reweighing + SVM

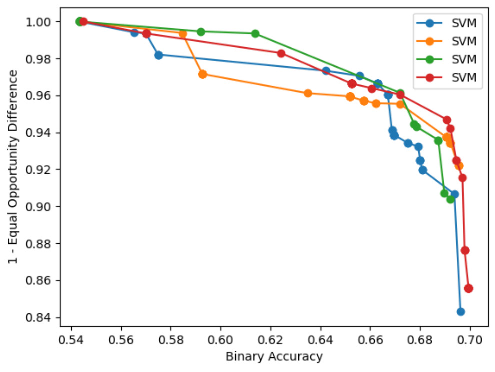


c) Disparate Impact Remover + SVM

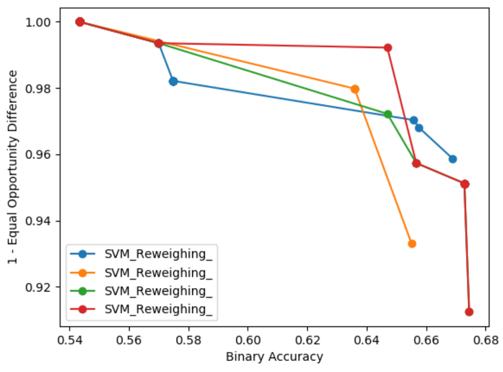


d) Optimized Pre-processing + SVM

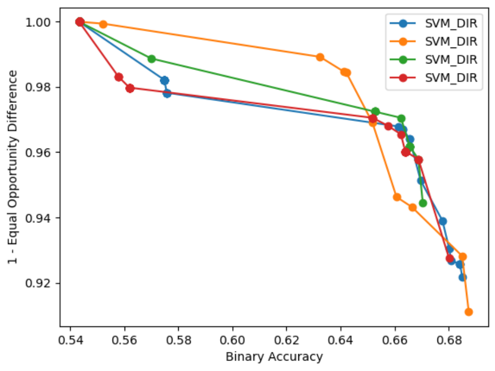
Figure 5.2: *Pareto fronts for Binary Accuracy vs. Theil Index on the COMPAS dataset. Four runs for each bias mitigation approach.*



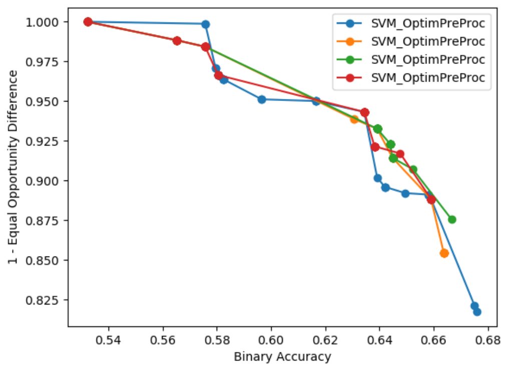
a) No mitigation approach, just baseline SVM



b) Reweighing + SVM

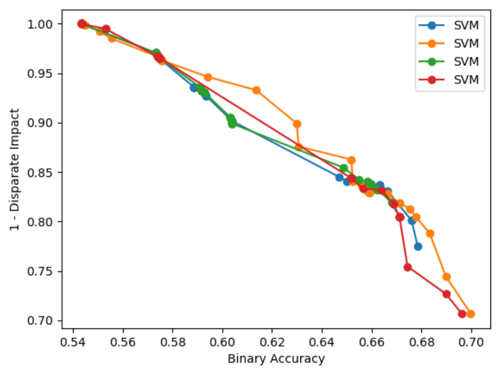


c) Disparate Impact Remover + SVM

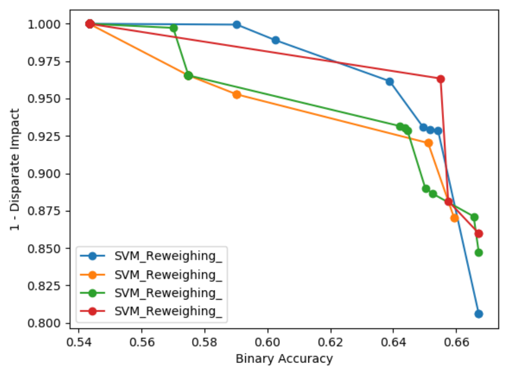


d) Optimized Pre-processing + SVM

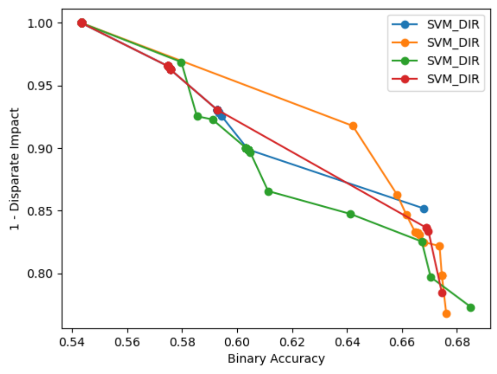
Figure 5.3: *Pareto fronts for Binary Accuracy vs. Equal Opportunity on the COMPAS dataset. Four runs for each bias mitigation approach.*



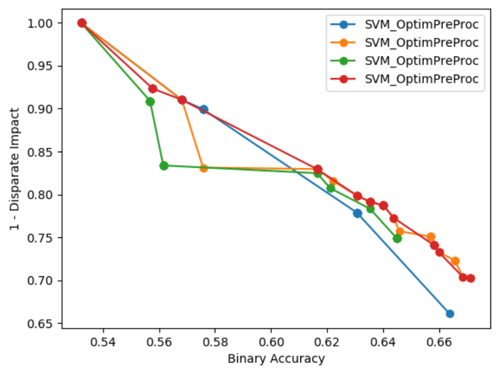
a) No mitigation approach, just baseline SVM



b) Reweighing + SVM



c) Disparate Impact Remover + SVM



d) Optimized Pre-processing + SVM

Figure 5.4: *Pareto fronts for Binary Accuracy vs. Disparate Impact on the COMPAS dataset. Four runs for each bias mitigation approach.*

Next, for each of the five scenarios, we chose one frontier for each algorithm for comparison between the approaches. When deciding which frontier to choose, we tried to pick one that dominates the others as much as possible, and if two frontiers dominate each other in separate parts along the x-axis, we favored the frontier that tends to dominate the other for higher accuracies.

Figure 5.5 shows the four frontiers plotted together for scenario 1: Binary Accuracy vs. Statistical Parity Difference.

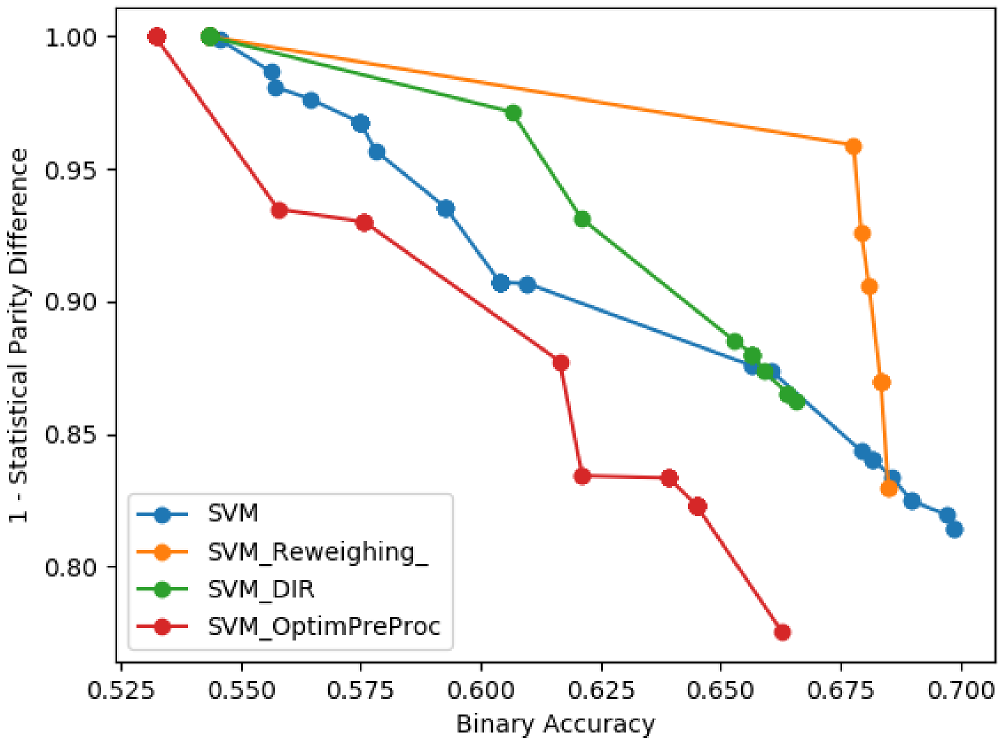


Figure 5.5: *Pareto fronts for the COMPAS dataset for all algorithms for scenario 1: Accuracy vs. Statistical Parity Difference*

The plots in figure 5.5 show that the Reweighing+SVM option outperforms all the other strategies in terms of fairness. The figure also shows that the bias mitigation approaches all lead to a clear accuracy penalty, just like the theory we reviewed in section 3.3 predicts. The highest accuracy achieved in this run of the Reweighing algorithm is around 0.685, and at that accuracy, the fairness is no better than what the no-mitigation approach achieves at the same level of accuracy. The no-mitigation approach achieves a maximum accuracy of around 0.7, but if one is willing to reduce the prediction accuracy to around 0.68, one can achieve a very large fairness enhancement by using the Reweighing algorithm. This is in line with published findings [43] from the researchers behind the **aif360** tool, who have found that Reweighing and the post-processing method called Reject option classification were the best fairness tools in their toolkit for the COMPAS dataset, and that the Reweighing mitigation method is able to achieve good fairness enhancement on the COMPAS data without much penalty in accuracy. Our finding is also in perfect harmony with the findings from the Haas study [35], where they found similar effects between fairness and AUC.

The plots in figure 5.5 also show that the Disparate Impact Remover approach succeeds in increasing fairness with respect to Statistical Parity compared to the baseline SVM, but at a slightly higher accuracy cost. One needs to drop accuracy to about 0.625 in order to get good fairness enhancement with this method. Optimized Pre-processing, however, makes the outcomes less fair than the no-mitigation approach, indicating that this mitigation approach is not a viable choice for this particular fairness metric and this particular dataset and prediction task.

Interestingly, given that Statistical Parity Difference is defined as the difference in the probability of favorable outcomes between the protected and non-protected groups, if our model simply accepts everyone, there will be no discrimination according to this metric. It can be observed that most runs of the algorithms in scenario 1 generate Pareto fronts with one solution with a perfect fairness score of 1.0, and an accompanying accuracy value around 54-55%. For the COMPAS dataset, approximately 55% of the labels are positive labels (the exact number depends on the training/test-split), which means that very likely, these "perfect fairness" solutions were achieved by the model simply predicting everyone as belonging to the positive class. This idea is strengthened by the fact that none of the algorithms achieve perfect fairness with respect to the Theil index (see figure 5.6).

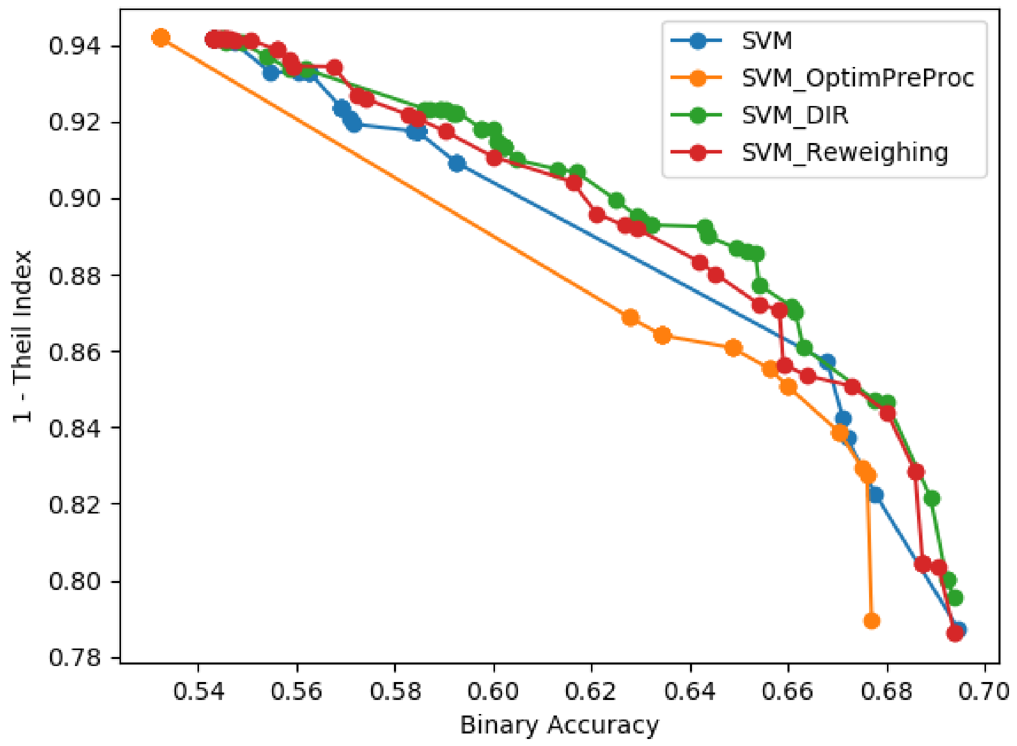


Figure 5.6: *Pareto fronts for the COMPAS dataset for all algorithms for scenario 2: Accuracy vs. Theil Index*

For our individual fairness metric, Theil Index, the experimental results again verify the theory that fairness enhancement comes at an accuracy cost. This can be seen in figure 5.6, where the Pareto fronts for the four algorithms are plotted against each other. Two of the three mitigation approaches; Reweighing + SVM, and Disparate Impact Remover + SVM, are able to produce better fairness than the no-mitigation, SVM-only approach, but only at accuracies that are lower than the maximum accuracy that can be achieved by the no-mitigation approach. Optimized Pre-processing + SVM, however, again performs worse than the no-mitigation approach with respect to fairness, just as in scenario 1. Notably, fairness enhancements are not as large for the Theil metric as they were for the Statistical Parity metric in scenario 1, and the Pareto fronts are overlapping and much more similar. This is in line with the results obtained in the Haas study [35], and indicates that individual fairness is harder to achieve on this dataset, or perhaps even in general.

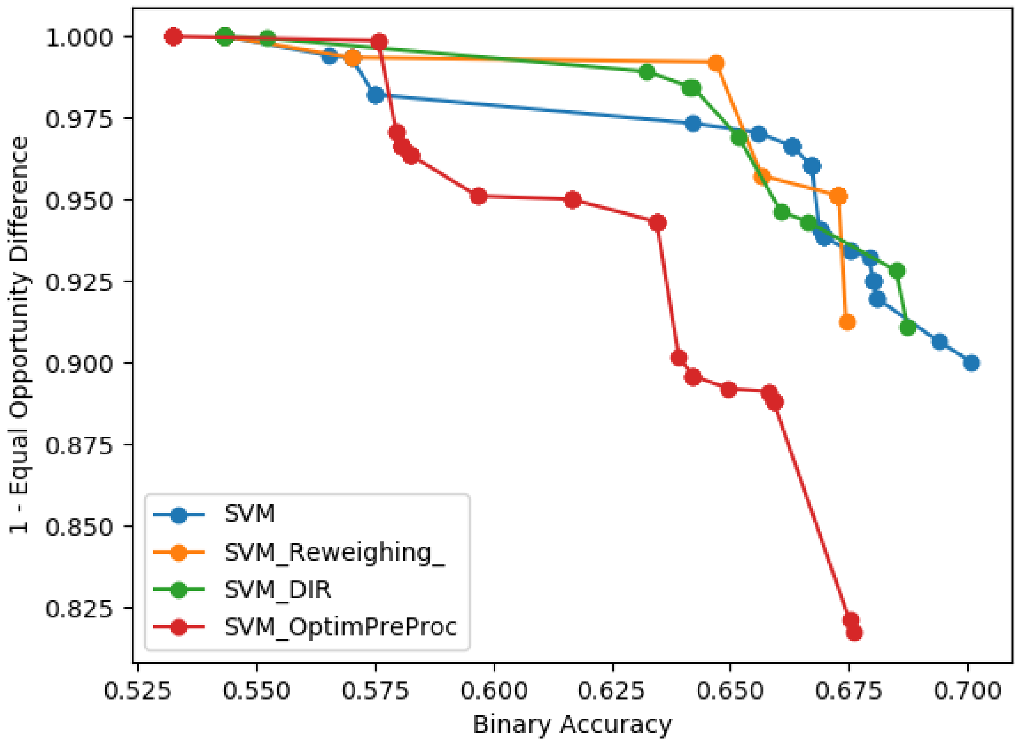


Figure 5.7: *Pareto fronts for the COMPAS dataset for all algorithms for scenario 3: Accuracy vs. Equal Opportunity Difference*

Results for the four mitigation approaches for Accuracy vs. Equal Opportunity Difference is shown in figure 5.7. Again the results show a clear accuracy penalty when applying bias mitigation strategies. Against this fairness metric, it seems that one needs to reduce accuracy to 0.65 before the Reweighing and the Disparate Impact Remover approaches clearly outperform the no-mitigation approach in terms of fairness. The no-mitigation approach again is able to reach an accuracy of about 0.7. We notice that for the first time in our experiments so far the Optimized Pre-processing approach is able to produce fairness-enhancement compared to the baseline SVM, but it comes at a hefty accuracy cost: one needs to drop accuracy all the way to about 0.58 to achieve this, making the algorithm only slightly more accurate than a coin toss.

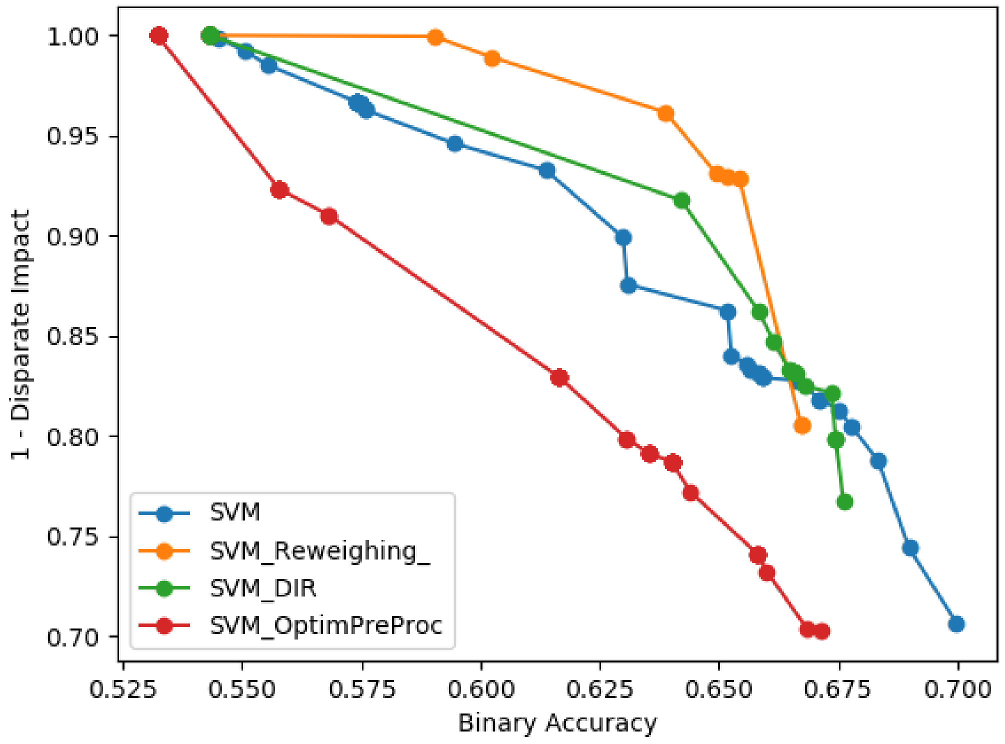


Figure 5.8: *Pareto fronts for the COMPAS dataset for all algorithms for scenario 4: Accuracy vs. Disparate Impact*

Figure 5.8 shows the plots for all algorithms for scenario 4: binary accuracy vs. Disparate Impact. The plots show improved fairness from the Reweighing approach and from the Disparate Impact Remover approach if one is willing to accept a reduction in accuracy to around 0.67, from the 0.7 accuracy that the pure SVM approach is able to achieve. Somewhat surprisingly, the Disparate Impact Remover algorithm did not achieve the best fairness results even though the fairness metric is precisely Disparate Impact. We do not know why that is so, but it is a testament to the strength of the Reweighing algorithm for the COMPAS dataset. Like in scenario 1, the Optimized Pre-processing approach makes the fairness worse than the no-mitigation approach, making it an unsuitable mitigation approach for this particular dataset & fairness metric.

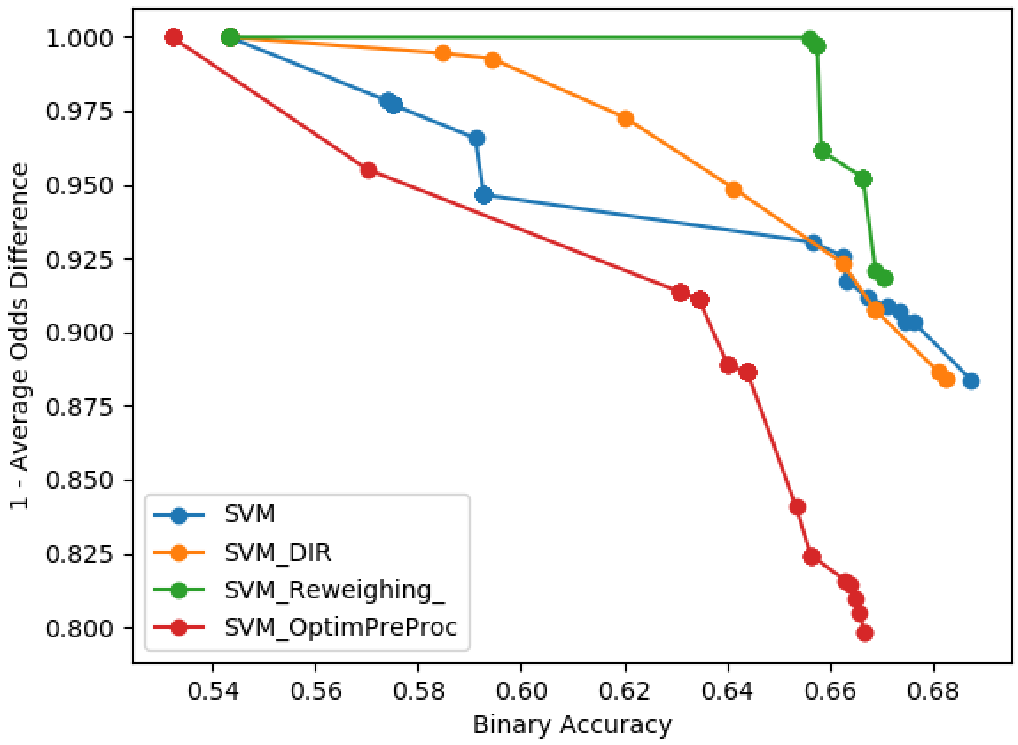


Figure 5.9: *Pareto fronts for the COMPAS dataset for all algorithms for scenario 5: Accuracy vs. Average Odds Difference*

Results for the final scenario, Accuracy vs. Average Odds Difference, is shown in figure 5.9. The plots show more of what we have already seen: the Reweighing algorithm outperforming the others, the Optimized Pre-processing algorithm being useless, and the Reweighing algorithm and the Disparate Impact Remover algorithm both outperforming the no-mitigation approach at a slight accuracy penalty. This particular fairness metric, defined as the average of difference in false positive rates and true positive rates between protected and non-protected groups, is particularly interesting for the COMPAS dataset, as the criticism towards the COMPAS system is that it produce more false positives for blacks (the protected group) than for the non-protected group.

All our results show that trade-offs between fairness and accuracy can clearly be demonstrated, and these trade-offs are visualized in figures 5.5 - 5.9. All five scenarios illustrate that if one is willing to accept slightly lower accuracy, very large gains in fairness can sometimes be achieved.

Our investigations indicate that our approach, as laid out in chapter 4, works as intended. All the plots we have generated show Pareto fronts that can be helpful when making decisions about what type of algorithm / mitigation method and metrics to use when building algorithmic decision systems. By applying the Haas framework [35] on other metrics and datasets than what they used in their study, we have added validity to their results. Even though the Haas framework can perhaps not be said to be a state of the art framework for investigating fairness-accuracy trade-offs, it is the closest to such that we have found, and our results clearly show that their framework is a useful tool when investigating fairness-accuracy trade-offs. Our investigations also support previous work about fairness-accuracy trade-offs, e.g., by Menon and Williamson [34], Zliobaite [96], and Corbett-Davies et al. [31].

With that, we have answered research question 3 (see section 1.3), and hence we have now concluded the task of answering all three research questions in our thesis. In the next chapter, chapter 6, we will present our concluding thoughts, as well as ideas for future work.

Chapter 6

Discussion

This chapter will present our concluding thoughts. Section 6.1 summarizes our results, and section 6.2 discusses our contributions, while section 6.3 provides exciting avenues for future research.

6.1 Summary

The field of fair ML continues to be a hot topic in research and the popular press alike. During the time of this writing (May 2020), for example, the US Department of Justice is being criticized for using a machine learning risk assessment tool called PATTERN, known for racial bias, to decide which prisoners are sent home early from US prisons to reduce population size due to COVID-19 concerns [103], and news just broke that even medical diagnosis systems are subject to biases that lead to adverse outcomes for certain groups [20].

Detecting bias in algorithmic systems has proven to be hard, and it is usually not done until well after systems have been deployed in real-life settings for some time. Mitigating known biases is also hard, and usually comes at the cost of reducing the accuracy of algorithmic systems. On that backdrop, we set out to answer three research questions (see section 1.3), intended to contribute to the understanding of causes of bias and unfairness in classifiers, and to contribute to the understanding of how current state of the art bias mitigation methods affect algorithmic performance.

To answer research question 1, which asks what bias in algorithms and datasets cause unfair outcomes in machine learning systems, we reviewed literature and proposed a taxonomy of bias types. The results of this work were presented in section 3.1 and 3.2. The two sections collectively answer research question 1.

Research question 2 asked what the state of the art methods for investigating trade-offs between fairness and accuracy in machine learning classification systems are. We performed a literature review that (a) showed that such trade-offs do in fact exist, and (b) looked for state-of-the-art frameworks to investigate the trade-offs further (see section 3.3). We failed to find a clear, agreed-upon state-of-the-art framework, but we decided that the Haas framework [35] is a good candidate. In order to confirm or disprove the usefulness of the Haas framework, we implemented it, designed a set of experiments to test it, and applied it on a different dataset and with different bias mitigation methods and against different fairness metrics than Haas used in their paper. These experiments intended to help us answer research question 3, which asked whether we can apply the method(s) or framework(s) identified in RQ2 on new problem settings, and what conclusions can we draw from that? Our results and analysis, along with our answer to research question 3, were presented in section 5.3.

6.2 Contributions

Solving the problem of bias and unfairness in machine learning requires an interdisciplinary approach, and papers related to the field are published by researchers from diverse disciplines such as law, ethics, philosophy, statistics, machine learning, and social sciences. Each of these disciplines brings their own nomenclature, and consequently, the fair ML literature is somewhat chaotic. Fair ML terminology is not aligned, sometimes even conflicting, with definitions that are too broad to be useful, and papers that often lack the necessary shared terminology to be understandable outside of specialty fields. Our proposed taxonomy of common bias types may contribute to an increased understanding of the problem state. This is one of the most important contributions of this paper. The attempts that have been made in previous papers to present a full taxonomy of fairness-related bias types have, in our opinion, been lacking, and so our approach to bias, which is presented in section 3.2, fills a gap in the ML fairness literature.

A thorough understanding of fairness-related bias is essential for several reasons. In recent years, several open-source libraries and tool-kits for detecting and mitigating bias in algorithms have been made public. While this is a good thing, over-reliance on libraries and de-biasing tools may give a false sense of security that a system is fair, when in fact, all that is achieved is a reasonable certainty that the system does not violate the specific fairness metrics that the tool looks for. Therefore, developers still need a good understanding of fairness and bias issues.

Also, it is a known fact that de-biasing a system after it has been trained is hard, often leading to bad trade-offs between fairness and accuracy [104]. This has been confirmed in our own experiments in this thesis. Therefore, attempts should go into making models bias-free from the start of the development process, similarly to what is common in related fields such as computer security or privacy. To understand, detect, and mitigate unfairness in algorithmic systems, AI developers need a good understanding of the concept of bias, and how unfairness can enter systems.

Another contribution of the thesis is to add support to leading theories about the trade-offs between fairness and accuracy in machine learning models (see section 3.3). Also, to add validity to the usefulness of the Haas framework [35] for evaluating trade-offs between various fairness and performance metrics. It is well documented that there is a reproducibility crisis in AI research, so being able to reproduce results from the Haas study is a research contribution in and of itself. In addition, our experiments extended their results on new datasets, other mitigation methods, and more fairness metrics, thus allowing us to cement their findings further. We believe that our results show that engineers and decision-makers should be able to evaluate different approaches for bias mitigation against each other by using MOO, and to use that approach to identify which models and methods work best for their use case, choice of fairness metrics, and datasets. Given the extremely high impact on peoples' lives that many algorithmic decision-making systems have, such investigations should be well worth the effort.

The Haas paper that we have modeled our experiments after was not published until December 2019, and we did not become aware of it right away. Therefore, the direction our applied experiments took was determined rather late in the master's project. Additionally, the applied experiments were somewhat hindered by the covid-19 situation that led to the closing down of the University in mid-March. Given more time and access to better computing power, we would have liked to extend our investigations to other ML learning algorithms, as well as perform our experiments on more datasets, with more bias mitigation methods. Nevertheless, we feel that the experiments we did perform led to clear outcomes, and we are pleased with the theoretical understanding of the problem area that we have gained during the last year, and that we have hopefully been able to express parts of in this thesis.

6.2 Future Work

At the end of the previous section, we mentioned a few ways that our research might be extended in future work. Here we will propose some exciting avenues for future work that are not directly related to the work in this thesis, but which we find interesting based on our current understanding of the field.

Like we have shown, de-biasing a system after it has been trained is hard and will often lead to bad trade-offs between fairness and accuracy. Instead, attempts should go into making systems bias-free from the start of the development process instead. A very exciting avenue for future research is, therefore, to look into how fairness and non-discrimination can be included as part of the objectives that machine learning models are evaluated on during training.

Even if ML models are trained with fairness in mind, it is unlikely that all bias issues can be anticipated in advance, so it is still important to test systems thoroughly before they are employed. Adversarial testing, both pre- and post-launch, might be an interesting avenue for future research [105].

Finally, our third suggestion for future research is related to causal models, based on the work of Judea Pearl. Part of the problem with unfairness in algorithmic decision-making systems is that many such systems turn correlative insights into causal scoring mechanisms, without there being any true causation behind the correlation. For example, in recidivism prediction, an algorithm may find correlations in historical crime data between where a person lives and recidivism risk, but the system does not know whether the person's address actually has a causal effect on crime, yet may base its predictions on precisely that. Several papers have been published during the last couple of years that investigate causal, or so-called counterfactual, fairness, and we believe this is one of the most promising roads forward for the field.

Bibliography

[1] M. Hardt, E. Price, and N. Srebro, “Equality of Opportunity in Supervised Learning,” *30th Conf. Neural Inf. Process. Syst. (NIPS 2016), Barcelona, Spain*, 2016.

[2] L. T. Liu, S. Dean, E. Rolf, M. Simchowitz, and M. Hardt, “Delayed impact of fair machine learning,” *35th Int. Conf. Mach. Learn. ICML 2018*, vol. 7, pp. 4929–4958, 2018.

[3] J. Angwin, J. Larson, S. Mattu, and L. Kirchner, “Machine Bias — ProPublica,” *ProPublica*, 2016. [Online]. Available: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. [Accessed: 18-Oct-2019].

[4] A. W. Flores *et al.*, “False Positives, False Negatives, and False Analyses: A Rejoinder to ‘Machine Bias: There’s Software Used Across the Country to Predict Future Criminals. And it’s Biased Against Blacks.’ The authors wish to thank.”

[5] “When Algorithms Discriminate - The New York Times.” [Online]. Available: https://www.nytimes.com/2015/07/10/upshot/when-algorithms-discriminate.html. [Accessed: 21-Nov-2019].

[6] L. Hu and Y. Chen, “A Short-term Intervention for Long-term Fairness in the Labor Market,” in *WWW ’18: Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1389–1398.

[7] D. Ensign, S. A. Friedler, S. Neville, C. Scheidegger, and S. Venkatasubramanian, “Runaway Feedback Loops in Predictive Policing,” *Proc. Mach. Learn. Res. 811–12, 2018 Conf. Fairness, Accountability, Transpar.*, Jun. 2018.

[8] A. Vestby and J. Vestby, “Machine Learning and the Police: Asking the Right Questions,” *Polic. A J. Policy Pract.*, vol. 0, no. 0, pp. 1–15, 2019.

[9] P. Rajpurkar *et al.*, “CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning,” Nov. 2017.

[10] P. Rajpurkar *et al.*, “Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists,” *PLOS Med.*, vol. 15, no. 11, p. e1002686, Nov. 2018.

[11] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, “Dissecting racial bias in an algorithm used to manage the health of populations,” *Science (80-.).*, vol. 366, no. 6464, pp. 447–453, Oct. 2019.

[12] P. Molnar and L. Gill, “Bots at the Gate: A Human Rights Analysis of Automated Decision-Making in Canada&#39;s Immigration and Refugee System,” *Int. Hum. Rights Progr. Citiz. Lab, Univ. Toronto*.

[13] L. Seyyed-Kalantari, G. Liu, M. Mcdermott, and M. Ghassemi, “CheXclusion: Fairness gaps in deep chest X-ray classifiers,” 2020.

[14] S. Barocas, M. Hardt, and A. Narayanan, “Fairness in Machine Learning Limitations and Opportunities,” 2019.

[15] S. Danziger, J. Levav, L. Avnaim-Pesso, and D. Kahneman, “Extraneous factors in judicial decisions,” *PNAS*, vol. 108, no. 17, pp. 6889–6892, 2011.

[16] B. Friedman and H. Nissenbaum, “Bias in Computer Systems,” *ACM Trans. Inf. Syst.*, vol. 14, no. 3, pp. 330–347, 1996.

[17] D. Leslie, “Understanding artificial intelligence ethics and safety: A guide for the responsible design and implementation of AI systems in the public sector,” *Alan Turing Institute. https//doi.org/10.5281/zenodo.3240529*, 2019.

[18] A. Chouldechova and A. Roth, “The Frontiers of Fairness in Machine Learning,” Oct. 2018.

[19] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, “Dissecting racial bias in an algorithm used to manage the health of populations.,” *Science*, vol. 366, no. 6464, pp. 447–453, 2019.

[20] A. J. Larrazabal, N. Nieto, V. Peterson, D. H. Milone, and E. Ferrante, "Gender imbalance in medical imaging datasets produces biased classifiers for computer-aided diagnosis," *Proc. Natl. Acad. Sci.*, vol. 117, no. 23, p. 201919012, May 2020.

[21] A. Chouldechova, "Fair Prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments," *Big Data*, vol. 5, no. 2, pp. 153–163, Jun. 2017.

[22] "Google Photos labeled black people 'gorillas,'" 2015. [Online]. Available: https://eu.usatoday.com/story/tech/2015/07/01/google-apologizes-after-photos-identify-black-people-as-gorillas/29567465/. [Accessed: 28-Oct-2019].

[23] "Apple's credit card is being investigated for discriminating against women - The Verge," 2019. [Online]. Available: https://www.theverge.com/2019/11/11/20958953/apple-credit-card-gender-discrimination-algorithms-black-box-investigation. [Accessed: 21-Nov-2019].

[24] D. Danks and A. J. London, "Algorithmic Bias in Autonomous Systems," no. January, 2018.

[25] A. Rajkomar, M. Hardt, M. D. Howell, G. Corrado, and M. H. Chin, "Ensuring fairness in machine learning to advance health equity," *Ann. Intern. Med.*, vol. 169, no. 12, pp. 866–872, Dec. 2018.

[26] S. Yao and B. Huang, "Beyond Parity: Fairness Objectives for Collaborative Filtering," *Adv. Neural Inf. Process. Syst.*, vol. 2017-December, pp. 2922–2931, May 2017.

[27] A. Caliskan, J. J. Bryson, and A. Narayanan, "Semantics derived automatically from language corpora contain human-like biases," *Science (80-.).*, vol. 356, no. 6334, pp. 183–186, 2017.

[28] F. Kamiran, T. Calders, and M. Pechenizkiy, "Discrimination aware decision tree learning," *Proc. - IEEE Int. Conf. Data Mining, ICDM*, pp. 869–874, 2010.

[29] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," in

Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, vol. 2015-August, pp. 259–268.

[30] B. Fish, J. Kun, and Á. D. Lelkes, “A confidence-based approach for balancing fairness and accuracy,” in *16th SIAM International Conference on Data Mining 2016, SDM 2016*, 2016, pp. 144–152.

[31] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq, “Algorithmic decision making and the cost of fairness,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, vol. Part F1296, pp. 797–806.

[32] J. Kleinberg, S. Mullainathan, and M. Raghavan, “Inherent trade-offs in the fair determination of risk scores,” in *Leibniz International Proceedings in Informatics, LIPIcs*, 2017, vol. 67.

[33] J. Kleinberg, “Inherent Trade-Offs in Algorithmic Fairness,” in *Abstracts of the 2018 ACM International Conference on Measurement and Modeling of Computer Systems - SIGMETRICS ’18*, 2018, pp. 40–40.

[34] A. K. Menon and R. C. Williamson, “The Cost of Fairness in Binary Classification,” in *Proceedings of the 1st Conference on Fairness, Accountability and Transparency, PMLR 81:107-118, 2018*, 2018, vol. 81, pp. 1–12.

[35] C. Haas, “The Price of Fairness - A Framework to Explore Trade-Offs in Algorithmic Fairness,” *Int. Conf. Inf. Syst. 2019*, pp. 1–17, 2019.

[36] B. Mittelstadt, “AI Ethics -- Too Principled to Fail?,” Jun. 2019.

[37] “Many Experts Say We Shouldnt Worry About Superintelligent AI. They’re Wrong - IEEE Spectrum,” 2019. [Online]. Available: https://spectrum.ieee.org/computing/software/many-experts-say-we-shouldnt-worry-about-superintelligent-ai-theyre-wrong. [Accessed: 23-Oct-2019].

[38] S. Russell, *Human Compatible - Artificial Intelligence and the Problem of Control*. Viking Press, 2019.

[39] P. Domingos, *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*. Allen Lane, 2015.

[40] "Google's Go-playing AI still undefeated with victory over world number one | DeepMind | The Guardian," 2017. [Online]. Available: https://www.theguardian.com/technology/2017/may/25/alphago-google-ai-victory-world-go-number-one-china-ke-jie. [Accessed: 08-Nov-2019].

[41] "AI detects skin cancers with more accuracy than dermatologists," 2018. [Online]. Available: https://www.radiologybusiness.com/topics/artificial-intelligence/ai-detects-skin-cancers-more-accuracy-dermatologists. [Accessed: 08-Nov-2019].

[42] A. Arora, *A multidisciplinary survey on discrimination analysis*, vol. 00. 2005.

[43] R. K. E. Bellamy *et al.*, "AI Fairness 360: An Extensible Toolkit for Detecting and Mitigating Algorithmic Bias," *IBM J. Res. Dev.*, pp. 1–1, Sep. 2019.

[44] L. Alexander, "What Makes Wrongful Discrimination Wrong? Biases, Preferences, Stereotypes, and Proxies," 1992.

[45] P. Gajane and M. Pechenizkiy, "On Formalizing Fairness in Prediction with Machine Learning," Oct. 2017.

[46] D. Pedreshi, S. Ruggieri, and F. Turini, "Discrimination-aware data mining," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 560–568, 2008.

[47] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel, "Fairness through awareness," in *ITCS 2012 - Innovations in Theoretical Computer Science Conference*, 2012, pp. 214–226.

[48] T. Calders and S. Verwer, "Three naive Bayes approaches for discrimination-free classification," in *Data Mining and Knowledge Discovery*, 2010, vol. 21, no. 2, pp. 277–292.

[49] M. Srivastava, H. Heidari, and A. Krause, "Mathematical Notions vs. Human Perception of Fairness: A Descriptive Approach to Fairness for Machine Learning," Feb. 2019.

[50] S. Mitchell, E. Potash, S. Barocas, A. D', A. G. Research, and K. Lum, "Prediction-Based Decisions and Fairness: A Catalogue of Choices, Assumptions, and Definitions," 2019.

[51] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth, "Fairness in Criminal Justice Risk Assessments: The State of the Art," *Sociol. Methods Res.*, 2017.

[52] A. Chouldechova, "Fair prediction with disparate impact: A study of bias in recidivism prediction instruments," *Artif. Intell. Law*, vol. 25, pp. 5–27, Oct. 2016.

[53] M. Kusner, J. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 4067–4077, 2017.

[54] T. Speicher *et al.*, "A unified approach to quantifying algorithmic unfairness: Measuring individual & group unfairness via inequality indices," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2018, pp. 2239–2248.

[55] P. Domingos, "A Few Useful Things to Know About Machine Learning," vol. 55, no. 10, 2012.

[56] "Why facial recognition's racial bias problem is so hard to crack - CNET," 2019. [Online]. Available: https://www.cnet.com/news/why-facial-recognitions-racial-bias-problem-is-so-hard-to-crack/. [Accessed: 05-Nov-2019].

[57] "Facial recognition AI from Amazon, Microsoft, and IBM misidentifies trans and non-binary people — Quartz," 2019. [Online]. Available: https://qz.com/1726806/facial-recognition-ai-from-amazon-microsoft-and-ibm-misidentifies-trans-and-non-binary-people/. [Accessed: 05-Nov-2019].

[58] M. Veale, M. Van Kleek, and R. Binns, "Fairness and accountability design needs for algorithmic support in high-stakes public sector decision-making," in *Conference on Human Factors in Computing Systems - Proceedings*, 2018, vol. 2018-April.

[59] M. Mitchell *et al.*, "Model cards for model reporting," *FAT* 2019 - Proc. 2019 Conf. Fairness, Accountability, Transpar.*, no. Figure 2, pp. 220–229, 2019.

[60] "AI Now 2017 Report," 2017.

[61] S. Barocas, M. Hardt, and A. Narayanan, "Fairness and machine learning," 2019. [Online]. Available: https://fairmlbook.org/. [Accessed: 13-Nov-2019].

[62] R. Mehrotra, A. Anderson, F. Diaz, A. Sharma, H. Wallach, and E. Yilmaz, "Auditing Search Engines for Differential Satisfaction Across Demographics," May 2017.

[63] S. Corbett-Davies and S. Goel, "The Measure and Mismeasure of Fairness: A Critical Review of Fair Machine Learning," Jul. 2018.

[64] S. Barocas and A. Selbst, "Big Data ' S Disparate Impact," *Law Rev*, vol. 671, pp. 671–732, 2016.

[65] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork, "Learning fair representations," *30th Int. Conf. Mach. Learn. ICML 2013*, vol. 28, no. PART 2, pp. 1362–1370, 2013.

[66] F. P. Calmon, D. Wei, K. N. Ramamurthy, and K. R. Varshney, "Optimized Data Pre-Processing for Discrimination Prevention," Apr. 2017.

[67] F. Kamiran and T. Calders, "Data preprocessing techniques for classification without discrimination," 2012.

[68] B. H. Zhang, B. Lemoine, and M. Mitchell, "Mitigating Unwanted Biases with Adversarial Learning," in *AIES 2018 - Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 2018, pp. 335–340.

[69] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma, "Fairness-aware classifier with prejudice remover regularizer," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, vol. 7524 LNAI, no. PART 2, pp. 35–50.

[70] E. L. Celis, L. Huang, V. Keswani, and N. K. Vishnoi, "Classification with fairness constraints: A meta-algorithm with provable guarantees," in *FAT* 2019 - Proceedings of the 2019 Conference on Fairness, Accountability, and Transparency*, 2019, pp. 319–328.

[71] G. Pleiss, M. Raghavan, F. Wu, J. Kleinberg, and K. Q. Weinberger, "On Fairness and Calibration."

[72] F. Kamiran, A. Karim, and X. Zhang, "Decision Theory for Discrimination-aware Classification," 2012.

[73] H. Suresh and J. V. Guttag, "A Framework for Understanding Unintended Consequences of Machine Learning," 2019.

[74] M. De-Arteaga *et al.*, "Bias in Bios: A Case Study of Semantic Representation Bias in a High-Stakes Setting," Jan. 2019.

[75] A. Torralba and A. A. Efros, "Unbiased Look at Dataset Bias."

[76] A. Olteanu, C. Castillo, F. Diaz, and E. Kıcıman, "Social Data: Biases, Methodological Pitfalls, and Ethical Boundaries," *Front. Big Data*, vol. 2, Jul. 2019.

[77] R. Dobbe, S. Dean, T. Gilbert, and N. Kohli, "A Broader View on Bias in Automated Decision-Making: Reflecting on Epistemology and Dynamics," Jul. 2018.

[78] R. Baeza-Yates, "Data and algorithmic bias in the web," pp. 1–1, 2016.

[79] M. Gupta, A. Cotter, M. M. Fard, and S. Wang, "Proxy Fairness," Jun. 2018.

[80] S. Silva and M. Kenney, "Algorithms, Platforms, and Ethnic Bias: An Integrative Essay," 2018.

[81] T. Calders and I. Žliobaitė, "Why unbiased computational processes can lead to discriminative decision procedures," in *Studies in Applied Philosophy, Epistemology and Rational Ethics*, vol. 3, Springer International Publishing, 2013, pp. 43–57.

[82] S. A. Friedler, C. Scheidegger, and S. Venkatasubramanian, "On the (im)possibility of fairness," Sep. 2016.

[83] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A Survey on Bias and Fairness in Machine Learning," Aug. 2019.

[84] “Statistical Bias Types explained (with examples) - part1.” [Online]. Available: https://data36.com/statistical-bias-types-explained/. [Accessed: 06-Nov-2019].

[85] J. Buolamwini and T. Gebru, “Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification,” *Proceedings of Machine Learning Research 81:1–15, 2018 Conference on Fairness, Accountability, and Transparency*, 2018. [Online]. Available: http://proceedings.mlr.press/v81/buolamwini18a.html?mod=article_inline. [Accessed: 18-Oct-2019].

[86] “E. Racial Bias | Center on Privacy and Technology.” [Online]. Available: https://www.perpetuallineup.org/findings/racial-bias#footnote228_4k66fc0. [Accessed: 22-Nov-2019].

[87] A. Chouldechova and M. G’Sell, “Fairer and more accurate, but for whom?,” Jun. 2017.

[88] “Medical school is more female than male, for the first time in history — Quartz at Work.” [Online]. Available: https://qz.com/work/1161347/medical-school-is-more-female-than-male-for-the-first-time-in-history/. [Accessed: 07-Nov-2019].

[89] “71 prosent kvinnelige medisinstudenter ved siste opptak - Jobb og utdanning - Dagens Medisin.” [Online]. Available: https://www.dagensmedisin.no/artikler/2016/01/14/mer-enn-70-prosent-kvinnelige-medisinstudenter-ved-siste-opptak/. [Accessed: 07-Nov-2019].

[90] M. Wick, S. Panda, and J.-B. Tristan, “Unlocking Fairness: a Trade-off Revisited,” *33rd Conf. Neural Inf. Process. Syst.*, no. NeurIPS, pp. 1–10, 2019.

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

[92] T. Sun *et al.*, “Mitigating Gender Bias in Natural Language Processing: Literature Review,” Jun. 2019.

[93] T. Kamishima, S. Akaho, and J. Sakuma, “Fairness-aware learning through regularization approach,” in *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2011, pp. 643–650.

[94] H. Lakkaraju, J. Kleinberg, J. Leskovec, J. Ludwig, and S. Mullainathan, “The Selective Labels Problem: Evaluating Algorithmic Predictions in the Presence of Unobservables,” 2017.

[95] M. B. Zafar, I. Valera, M. G. Rodriguez, and K. P. Gummadi, “Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment,” in *26th International World Wide Web Conference, WWW 2017*, 2017, pp. 1171–1180.

[96] I. Zliobaite, “On the relation between accuracy and fairness in binary classification,” May 2015.

[97] D. Dua and C. Graff, “UCI Machine Learning Repository,” *Irvine, CA: University of California, School of Information and Computer Science*, 2019. [Online]. Available: http://archive.ics.uci.edu/ml/index.php. [Accessed: 01-Jun-2020].

[98] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II,” 2002.

[99] D. E. Goldberg, D. Mills, O. • Madrid, • San, J. • New, and Y. • Singapore, “Genetic Algorithms in Search, Optimization, and Machine Learning,” 1989.

[100] C. L. Huang and C. J. Wang, “A GA-based feature selection and parameters optimizationfor support vector machines,” *Expert Syst. Appl.*, vol. 31, no. 2, pp. 231–240, Aug. 2006.

[101] J. Larson, S. Mattu, L. Kirchner, and J. Angwin, “GitHub - propublica/compas-analysis: Data and analysis for ‘Machine Bias.’” [Online]. Available: https://github.com/propublica/compas-analysis. [Accessed: 02-Jun-2020].

[102] J. Dressel and H. Farid, “The accuracy, fairness, and limits of predicting recidivism,” *Sci. Adv.*, vol. 4, no. 1, Jan. 2018.

[103] “Letter to Attorney General Barr RE: The use of the PATTERN risk assessment in prioritizing release in response to the COVID-19 pandemic - The Leadership Conference on Civil and Human Rights.” [Online]. Available: https://civilrights.org/resource/letter-to-attorney-general-barr-re-the-use-of-the-

pattern-risk-assessment-in-prioritizing-release-in-response-to-the-covid-19-pandemic/. [Accessed: 21-Jun-2020].

[104] H. Jiang and O. Nachum, “Identifying and Correcting Label Bias in Machine Learning,” 2019.

[105] “10 things you should know about algorithmic fairness | ACM Interactions.” [Online]. Available: http://interactions.acm.org/archive/view/july-august-2019/10-things-you-should-know-about-algorithmic-fairness#R5. [Accessed: 08-Nov-2019].