UNIVERSITÉ DE LORRAINE | BIBLIOTHÈQUES UNIVERSITAIRES

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact bibliothèque : ddoc-theses-contact@univ-lorraine.fr
(Cette adresse ne permet pas de contacter les auteurs)

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4
Code de la Propriété Intellectuelle. articles L 335.2- L 335.10
http://www.cfcopies.com/V2/leg/leg_droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Hybrid processing for algorithmic fairness

Traitement hybride pour l'équité algorithmique

THÈSE

présentée et soutenue publiquement le 20 décembre 2022

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Guilherme Alves da Silva

Composition du jury

Président :	Anne Boyer	Pr. Université de Lorraine, LORIA
Rapporteurs :	Marie-Jeanne Lesot	MCF HDR, Sorbonne Université, LIP6
	Katharina Simbeck	Pr. HTW Berlin
Examinateur :	Fatiha Saïs	Pr. Université Paris Saclay, LISN
Invités :	Catuscia Palamidessi	DR Inria Saclay, LIX
	Francis Colas	CR Inria Nancy-Grand Est, LORIA
Directeurs :	Miguel Couceiro	Pr. Université de Lorraine, LORIA
	Amedeo Napoli	DR Emérite, CNRS, LORIA

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

It was a long journey until I finished this PhD thesis. I would like to thank several people I met during this period and without whom this thesis would not be possible. To begin with, I would like to express my gratitude to my supervisors. Thank you for accepting me as a PhD student!

Um agradecimento especial ao meu orientador principal, pelo apoio, energia e ousadia em mudar os rumos desta tese quando necessário durante esses 4 anos e 2 meses de doutoramento. Muito obrigado!

Je remercie aussi mon co-directeur de thèse pour avoir apporté son expérience, ses conseils et sa rigueur. Mais aussi pour avoir apporté son sens de l'humour particulier et de la musique. Merci !

I would like to thank all members of my thesis and follow-up committees. Thank you for accepting the invitation and also for the discussions and remarks. Special thanks go to the women on my thesis committee, particularly to the two researchers who wrote the reports of my thesis.

I also thank the interns with whom I worked on some implementations and experiments. Thank you for your energy and motivation. It helped me a lot, especially during the tough times.

Je remercie vivement le meilleur stagiaire[1] *de l'équipe Orpailleur avec qui j'ai eu le plaisir de travailler (entre 2020 et 2021). Cette thèse s'appuie sur une grande partie de notre travail commun et, sans aucun doute, il serait impossible de terminer ce document sans lui. Actuellement (2022), il prépare également une thèse et je ne peux pas manquer la chance d'enregistrer ici mes meilleurs souhaits pour sa thèse, sa carrière et sa vie également. Merci beaucoup, tu m'es très cher !*

I would like to thank the co-authors with whom I collaborated at different moments. In particular, those from the research team Comète in Paris. You helped me develop a broader view of research on algorithmic fairness and become a better researcher. Thank you!

During these years, I had the pleasure of meeting people in the lab that come from different parts of the world. Fortunately, some are (or were) members of the Orpailleur team to whom I register my special thanks. I spent such a great time with them at the lab and outside, and some became good friends. I also thank some current (and former) PhD students from other research teams of Loria for whom I am equally grateful for the time I spent with, that is, those from Capsid, Larsen, Gamble, Multispeech, Veridis, Semagramme, Optimist, Bird, Resist, and Biscuit teams. Thank you for the pleasant conversations during lunch, coffee breaks, and extra activities. Merci beaucoup ! Gracias! Ευχαριστώ! Спасибо! Thanmirth! Grazie mille! Vielen Dank!

During the second part of my PhD, I had the opportunity to teach. Thus, I would like to register my special thanks to some of the kind colleagues I had the privilege of working with at IDMC and ENSGSI.

Je tiens également à remercier ceux que j'ai rencontrés à Nancy pendant mes années de thèse, mais qui ne sont pas directement liés à la vie du labo. Ils sont ici ou là-bas. Ils ont joué un rôle important en rendant ma vie en France plus variée, plus douce et plus intéressante. Merci de m'avoir partagé votre culture et m'avoir intégré dans cette société.

Agradeço aos meus amigos brasileiros de Nancy (igualmente aos que estão espalhados pela França e Europa) que encontrei durante esses anos. Vocês foram responsáveis por trazer um pouco do nosso país fora das terras brasileiras. Valeu, galera!

[1] *quelqu'un qui a un grand "cœur" !*

Finalmente, agradeço aos meus pais e a minha família que do Brasil transmitiram o amor e o incentivo que me foi necessário para começar, continuar e terminar esta tese. Muito obrigado por sempre aquecerem meu coração apesar da distância. Eu os amo imensamente!

Naming each person individually in each paragraph would be unfeasible (and risky too, as I might forget someone). The people behind the names that are implicit[2] in each paragraph are able to identify themselves. They also know the impact they have on my PhD journey and how important they are to me. Therefore, their names are kept in a much better place than a paper or a digital file. That is, their names will remain imprinted on my memory and my heart and will remain so with me wherever I go.

"Amizade é dom que não se aprende, não
Bate de repente, quando a gente estende a mão
Bem maior que o amor
Muito mais que a paixão
Vem do fundo do coração"

(Boca Livre and Marcos Valle. Amizade.
In *Amizade*. Som Livre, 2014)

[2]The reader may notice that no names appear in this section.

To my family,
particularly my mother.

Contents

Introduction **1**

1 Algorithms and society . . . 1

1.1 From society to data (and biases) . . . 2

1.2 Bias and algorithmic fairness . . . 2

2 Research problems on algorithmic fairness . . . 3

3 Our approach in a nutshell . . . 4

4 Contributions . . . 5

5 Outline . . . 6

Partie I Fundamentals: fairness and explainability **9**

Chapter 1
Fairness notions

1.1 A general definition of fairness . . . 11

1.2 Fairness notions . . . 12

1.2.1 Group fairness . . . 13

1.2.2 Individual fairness . . . 17

1.3 Process fairness . . . 18

1.3.1 Improving process fairness . . . 19

1.4 Tensions between fairness notions . . . 20

Chapter 2
Unfairness mitigation

2.1 Pre-processors . . . 24

2.1.1 Reweighing [17] . . . 24

2.1.2 FairExp [75] . . . 24

2.1.3 FairBatch [74] . . . 25

2.2 In-processors . . . 25

2.2.1 Debiasing with adversarial learning [91] . . . 25
2.2.2 Exponentiated gradient and grid search reduction [2] . . . 25
2.2.3 Prejudice remover [50] . . . 26
2.2.4 AdaFair [46] . . . 26
2.3 Post-processors . . . 26
2.3.1 Threshold Optimizer [44] . . . 27
2.3.2 Reject option classification [49] . . . 27
2.3.3 Shifted Decision Boundary (SBD) [29] . . . 27
2.4 Hybrid processors . . . 27
2.4.1 Learning Fair Representations (LRF) [90] . . . 27
2.4.2 Fairness-Aware Ensemble (FAE) [45] . . . 28
2.5 The fairness-accuracy trade-off . . . 28

Chapter 3
Explanation methods

3.1 Explanation methods: properties and taxonomies . . . 31
3.2 Local post-hoc explanations . . . 33
3.2.1 Local Interpretable Model Agnostic Explanations (LIME) [72] . . . 34
3.2.2 SHapley Additive exPlanations (SHAP) [59] . . . 36
3.3 From local to global explanations . . . 37
3.4 Drawbacks and pitfalls of employing LIME and SHAP . . . 38
3.5 LIME and SHAP extensions . . . 39
3.6 PathExplain [47] . . . 40

Partie II Contributions **41**

Chapter 4
Mitigating unfairness through unawareness

4.1 Process fairness meets explanations . . . 44
4.2 Components of FixOut . . . 44
4.2.1 Fairness assessment . . . 45
4.2.2 Unfairness mitigation: the Ensemble Building component . . . 46
4.3 FixOut the algorithm . . . 49
4.4 Example . . . 51
4.5 Discussion . . . 51

Chapter 5
Mitigating unfairness on tabular data: the first application case of FixOut

5.1 Tabular datasets . . . 54
5.1.1 German . . . 55
5.1.2 Adult . . . 55
5.1.3 LSAC . . . 55
5.2 Experimental setup . . . 56
5.3 Experimental results . . . 56
5.3.1 Process fairness assessment . . . 56
5.3.2 Classification performance assessment . . . 58
5.3.3 Fairness metrics assessment . . . 59
5.4 Automating the choice of FixOut's parameter . . . 62
5.4.1 Selection of instances to assess fairness . . . 63
5.4.2 Experimental evaluation of Find-K . . . 64
5.5 Discussion and perspectives . . . 66

Chapter 6
Mitigating unfairness on textual data: FixOut's extensions

6.1 FixOut applied to models trained on textual data . . . 72
6.2 Experimental setup . . . 75
6.3 Experimental results . . . 75
6.3.1 Process fairness assessment . . . 75
6.3.2 Classification performance assessment . . . 76
6.4 FixOut for Neural Networks . . . 77
6.5 Discussion and perspectives . . . 79

Conclusion and perspectives **81**

Appendix A Correlation analysis **85**

Appendix B Further experiments I **87**

Appendix C Further experiments II **91**

Appendix D List of publications **95**

Glossary **97**

Glossary **97**

Notation **99**

Notation **99**

Résumé étendu **101**

Bibliography **107**

Introduction

1 Algorithms and society

Machine Learning (ML) models are increasingly present in decision support systems. The ML-based Decision Making (MLDM) systems and their algorithmic decisions are being used in a daily basis. However, ML models (and by extent MLDM) may be complex and opaque; the lack of transparency can hide inner workings that produce biased decisions. Also, recent studies have shown that MLDM systems to discriminate against minorities and unprivileged groups. These aspects raise several concerns given the critical impact that such prevalent MLDM systems may have on individuals or on society. Consequently, these problems shed light on the need for public trust in algorithmic decisions and more broadly in ML and Artificial Intelligence (AI).

Well-known examples include MLDM systems that predict credit card defaulters [38, 54], mortgage lending [55], criminal recidivism [7], and also multiple other systems which may impact government decisions. Revealing discriminating outcomes against minorities and unprivileged groups has also been done by investigations outside the academic community[3,4].

Figure 1 illustrates the interactions between society and MLDM systems. These systems are generally focused on optimizing and benefiting target sub-groups of society. However, the decisions taken by MLDM systems may affect the whole society in both ways positively and negatively, including non-users and non-target sub-groups.

Recently, governments have started regulating the AI-based systems market. For instance, in 2016, the European Union (EU) enforced the General Data Protection Regulation (GDPR)[5] that gives users the right to understand the inner workings of MLDM systems and to obtain explanations of their outcomes. Also in 2016, the Executive Office of the President of the United States released a report that surveys case studies and possible government policies to avoid discrimination in automated decisions [67]. More recently, the EU commission also released a

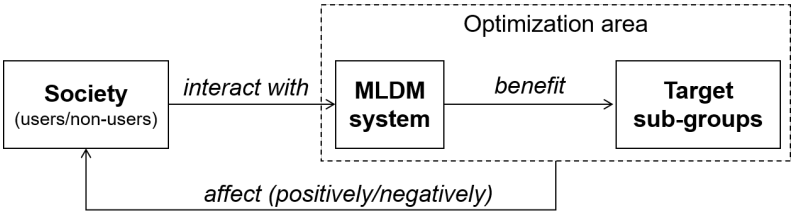


Figure 1: Society and MLDM system interactions.

[3] https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/idUSKCN1MK08G
[4] https://www.bbc.com/news/business-50365609
[5] https://gdpr-info.eu/

report that proposes actions to be taken based on the risk level of AI-based systems[6]. All these initiatives shed light on the need of fairness in order to reduce the negative societal impact of MLDM systems and, as a consequence, increase public trust in ML.

1.1 From society to data (and biases)

One of the main reasons to replace humans by MLDM systems is that human decisions are subjective while machines are supposed to make objective decisions. This reasoning hides an strong assumption that states the two following conditions hold: (1) we live in a world free from inequalities and (2) the data inputted into MLDM systems is free from biases. However, this assumption does not hold in reality. First, the actual world keeps social structures (e.g., social classes, religions, and laws) that perpetuate discrimination against sub-populations. Second, the data used to train ML models are often biased even if the collected data represents well the actual world; in addition, the data often suffers from measurement error and sampling bias. Hence, two different types of biases can be identified: *societal bias* and *statistical bias*.

In [63] these two types of biases are discussed and presented in the form of gaps among three main entities, namely: the ideal world ("World as it should and could be"), the actual world ("World as it is"), and the world according to data (see Figure 2). From the ideal world to the actual world we find the societal bias (non-statistical bias). The gap between these two first entities can be originated from retrospective injustice based on historical social structures. The second one is the statistical bias which can be originated from under-represented populations that come from the sampling approach (*sampling bias*) and systematic measurement error.

On a related research line, Friedler et al. introduce three metric spaces in order to relate issues of bias to discrimination and fairness. The first metric space is the construct space (or unobserved space) that contains the set of desired features (e.g. intelligence). Since we might not be able to measure the desired features, Friedler et al. also introduce the observed space that contains the set of measurable features (e.g. GPA score). They also introduce the decision space that contains the set of information that represent the outcomes (e.g. college admission or rejection) [32].

In addition, Friedler et al. also introduce two different worldviews , namely: WYSIWYG (What you see is what you get) and WAE (We're all equal). In WYSIWYG worldview the constructed space and the observed space are essentially considered the same, while in WAE there is no difference between individuals from different sub-populations based on sensitive (or salient) features (e.g. gender and ethnicity) [32].

1.2 Bias and algorithmic fairness

Bias plays an important role for ML models, since they are designed to have some bias that guide them in their target tasks. There is indeed no learning without bias. However, ML models have various biases that can be originated from multiple sources and some may not be desired [61]. If bias unexpectedly impacts society, in particular against minorities, which should rather be protected, then it characterizes a fairness issue [64].

Table 1 presents an example of two MLDM systems: one for assessing and predicting credit card defaulters and another for predicting whether a text contains hate speech. It also presents two possible biases for each system. On the one hand, a system for credit card default prediction is intended to be biased so that users who have a good credit payment history get a higher score than those who have not. Similarly, a system for hate speech detection is intended to be

[6] https://ec.europa.eu/commission/presscorner/detail/en/IP_21_1682

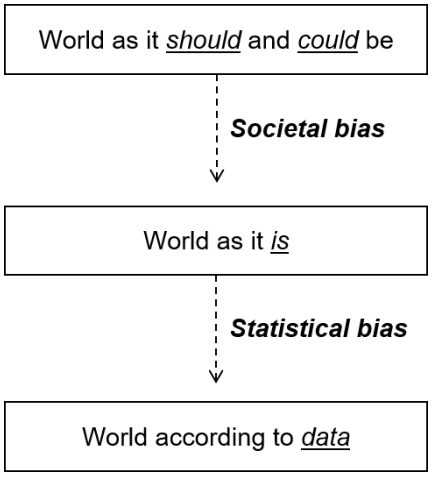


Figure 2: Two different types of biases from data [63].

Task	Intended Bias	Unintended Bias
Credit risk assessment	(good) *credit payment history* ↑	*ethnic minority* ↓
Hate speech prediction	(presence of) *offensive terms* ↑	*language variant* ↓

Table 1: Examples of intended and unintended biases on two different tasks.

biased such that messages that contain offensive terms get higher scores than those which have not. These are known as *intended bias*. On the other hand, the systems' outcomes should not rely on salient nor discriminatory features that carry prejudice towards unprivileged groups [37]. More precisely, the system that predicts credit card defaults should not give lower scores to users because they belong to a particular ethnicity compared to those who do not. In the same way, the system that detects hate speech should not give higher scores to messages that are written in a particular language variant compared to those messages that are not. These are known as *unintended biases* [27] and they have the potential to negatively impact society when present in MLDM systems.

Unintended biases and fairness issues have been found in a wide range of domains and data types. For instance, gender and ethnic discrimination were detected in online platforms for job applications [43]. Bias based on language variants was also found on approaches that detect abusive language, in which messages written in African-American English got higher toxicity scores than messages in Standard-American English [23].

This type of bias hidden in MLDM systems can automatize unfair decisions. Hence, these systems amplify discrimination against minorities due to unfair outcomes, which unveil the need for approaches that uncover and remove (or reduce) unintended bias. Assessing fairness and mitigating unfairness are the two main tasks that have motivated the growth of the research field called *algorithmic fairness* [8, 64].

2 Research problems on algorithmic fairness

In order to tackle the first main problem on algorithmic fairness, i.e. **assessing fairness**, several notions of fairness have been defined. Some notions focus on the outcomes of MLDM systems

and link to sensitive (or salient) features[7] – e.g. age, gender, ethnicity – through statistical measures [84, 87]. They are usually inspired by anti-discrimination efforts that aim to ensure that unprivileged groups – e.g. racial minorities – should be treated fairly. Although these notions have distinct semantics, the use of these definitions of fairness is criticized for being a reductionist understanding of fairness whose aim is basically to implement accept/not-accept reports [56].

Regarding the second main problem, i.e. **mitigating or removing unfairness**, mitigation approaches generally apply fairness interventions in specific steps of ML pipelines. More precisely, they usually change either (1) the data before training or (2) the optimization function of ML algorithms or (3) the algorithms' outputs in order to enforce fairer outcomes [71]. Recently, research on algorithmic fairness have been dedicated to explore combinations of different fairness interventions, e.g. modifying data before training along with the optimization function during the training phase. In this thesis, we arrange these new methods that combine fairness interventions in a fourth category called *fairness hybrid-processing*.

In a different research direction, ensemble-base methods for classification have achieved suitable results w.r.t. classification performance [15]. These methods take advantage of multiple and simple models as they rely on the idea that a set of models together outperforms single classifiers alone. Bagging and Boosting are among the most well-known ensemble-based approaches [14,89]. Recently, the idea of using ensembles has also been extended to the unfairness mitigation problem with results demonstrating that ensemble-based methods outperformed existing approaches [46]. However, those methods focus only on optimizing well-known fairness notions.

Once we try to mitigate unfairness, a tension between fairness and performance arises. When the focus is only on classification accuracy, this tension is also known as the *fairness-accuracy trade-off* [53]. In addition, unfairness mitigation approaches that optimize only traditional fairness notions help to reduce unfairness in ML models but ignore other perspectives on inequality and on societal impact.

In another pertaining framework, *process fairness* (or procedural fairness) is a subjective fairness notion which is centered on the process that leads to outcomes [37]. It quantifies the fraction of users that consider fair the use of a particular set of features. However, it is difficult and costly to estimate this information. Another way to assess process fairness is to estimate the dependence of ML models on sensitive features [12]. In this context, a category of explanation methods (explainers or explanators for short) can be employed, which connects explainability with process fairness. Explainers based on feature importance, e.g. Local Interpretable Model-agnostic Explanations (LIME) [72] and SHapley Additive exPlanations (SHAP) [59], output the contribution (or importance) of features for individual predictions. The idea behind the estimation of feature importance is that it can uncover unintended biases and by extent unfair outcomes. However, a very few works on process fairness are available in the literature.

In summary, existing approaches lack of: (i) model-agnostic and hybrid-processing approaches that tackle the fairness-accuracy trade-off and (ii) unfairness mitigation methods focus on improving process fairness.

3 Our approach in a nutshell

This thesis focuses on the fairness-accuracy trade-off problem since we are interested in reducing unintended biases without compromising classification performance. We thus propose ensemble-

[7] Also referred to as protected features.

based methods to find a good compromise between fairness and classification performance of ML models, in particular models for binary classification. In addition, these methods produce ensemble classifiers thanks to a combination of fairness interventions, which characterizes the fairness hybrid-processing approaches.

Hypothesis: Ensemble-based methods along with fairness hybrid-processing reduce unintended biases (and by extent mitigate unfairness) in ML models without compromising classification performance and without requiring awareness of the model type (model-agnostic).

In order to test our hypothesis, we propose FixOut (FaIrness through eXplanations and feature dropOut)[8], the human centered, model-agnostic framework that reduces classifiers' reliance on sensitive features without compromising their classification performance. It receives a classifier (pre-trained model), a dataset, a set of sensitive features, and an explanation method as input, and it outputs a new classifier that is less reliant on the sensitive features. To assess the reliance of a given pre-trained model on sensitive features, FixOut uses explanations to estimate the contribution of features to models' outcomes. If sensitive features are shown to contribute globally to models' outcomes, then the model is deemed unfair. In this case, a pool of fairer classifiers is built, which are then aggregated to obtain an ensemble classifier.

4 Contributions

The main contributions of this thesis are summarized as follows.

1. **Introduction of FixOut.** We propose a generic framework that tackles the problem of finding a good compromise between classification performance and process fairness, so-called FixOut. We perform an empirical study in order to evaluate the framework using different classifiers. This contribution can be decomposed into the following parts.

    - *Explanation methods.* FixOut goes further than existing methods by allowing the use of any explanation method that provides explanations in the form of feature importance.
    - *Aggregation functions.* We incorporate different aggregation functions in FixOut. One of these functions takes into consideration the contribution of sensitive features. It does a weighted aggregation by assigning high weights to those classifiers that were trained without sensitive features with high contributions.
    - *Fairness notions on top of* FixOut. We perform an empirical study in order to evaluate FixOut on well-known fairness notions, rather than considering only process fairness. The empirical study on fairness notions offers an interesting perspective of FixOut as it uncovers unfairness of models from the lens of well-known fairness notions. It shows that, in specific contexts, FixOut also helps to improve well-known fairness notions even though it is not designed for that purpose.

2. **Automating framework's parameter.** We proposed an algorithm that automatically selects the number of features that FixOut takes into account during the fairness assessment. This algorithm offers a useful tool in the scenarios of the presence of datasets with high dimensionality (number of features).

[8] https://fixout.loria.fr/

3. **FixOut for textual data.** We extended FixOut to models trained on textual data. The main originality of this version of FixOut resides in the way fairness through unawareness is employed, since the list of words to be taken into consideration is usually larger than the list of sensitive features in the previous setting.

4. **FixOut to model-specific settings.** We extended the framework to be model-specific. Particularly, we introduced a FixOut 's extension for neural networks (FixOut NN), due to recent highly visible successes of neural networks on several tasks.

5 Outline

This manuscript is organized as follows.

- **Chapter 1 [Fairness notions].** This chapter introduces the main fairness notions, their taxonomy, and their philosophical inspiration. In the same chapter, we also present the process fairness notion that is optimized in the framework FixOut. Finally, we highlight some tensions between fairness notions and discuss some impossibilities.

- **Chapter 2 [Unfairness mitigation].** This chapter overviews approaches that address the problem of mitigating unfairness in ML models. We start by the classical categories and then we propose a new category of unfairness mitigation approaches, hybrid-processing. We end this chapter by discussing the fairness-accuracy trade-off and some related work.

- **Chapter 3 [Explanation methods].** In this chapter, we present a taxonomy of explanation methods and we then focus on post-hoc explanations methods, since they can explain any model and are used in the framework FixOut. We also discuss some pitfalls on relying on explanations. Finally, we show how existing approaches that provide local explanations can be aggregated to obtain global explanations.

- **Chapter 4 [Mitigating unfairness through unawareness].** We present the first contribution of this thesis, the framework FixOut. In this chapter, we introduce the general idea of combining fairness through unawareness with an ensemble approach to mitigate unfairness without compromising classification performance. We detail the components of the framework, how we employ explanations to assess and improve process fairness, and the different aggregation functions for obtaining a single outcome from an ensemble of classifiers.

- **Chapter 5 [Mitigating unfairness on tabular data: the first application case of FixOut].** In this chapter, we detail the first application case of the general idea presented previously, the framework FixOut. We used an instantiation of FixOut and applied it on models trained on tabular data. Part of this chapter is based on one contribution from a conference article published in *AIST 2020* that evaluates the idea of using ensemble models for unfairness mitigation not only from the lens of process fairness but also from the perspective of the standard fairness notions [6].

- **Chapter 6 [Mitigating unfairness on textual data: FixOut's extensions].** In this chapter, we present the second application case of the general framework of FixOut applied to models trained on textual data. We detail how we adapt the concept of fairness through unawareness with ensemble models from a tabular data setting to a textual data setting. This chapter is mainly based on two contributions from a conference article published in

DSAA 2021 that introduces the version of FixOut for textual data and the method for automating the choice of FixOut's parameter [5].

- **Chapter Conclusion and perspectives.** We finally summarizes this manuscript and presents some research perspectives in the closing chapter. The list of publications is available in **Appendix D**.

Part I

Fundamentals: fairness and explainability

Chapter 1

Fairness notions

Contents

1.1 A general definition of fairness 11
1.2 Fairness notions 12
1.2.1 Group fairness 13
1.2.2 Individual fairness 17
1.3 Process fairness 18
1.3.1 Improving process fairness 19
1.4 Tensions between fairness notions 20

In order to address algorithmic fairness, various notions of fairness have been defined based on models' outcomes [60, 84, 87]. These metrics are described in Section 1.2. Another notion is process fairness which, instead of focusing on the final outcomes, is centered on the process that leads to the outcomes (classifications) [37]. The main idea is to assess the model's reliance on discriminatory or sensitive features, such as race, ethnicity, gender, or sexual orientation. This particular notion of fairness is presented in Section 1.3.

1.1 A general definition of fairness

In order to give a general definition of fairness, Firedler et al. [32] introduce three different metric spaces: construct, observed and decision.

The construct space ($\mathcal{CS}$) essentially consists of the desired or true features. That is, $\mathcal{CS} = (P, d_P)$ is the metric space where P is the set of true or desired features and d_P is the distance function that faithfully measure the closeness of two objects (or individuals) w.r.t. a specific task. The observed space ($\mathcal{OS}$) consists of the measurable features. In other words, the space is defined as $\mathcal{OS} = (\hat{P}, \hat{d})$, where $\hat{P}$ is the set of measurable features and $\hat{d}$ is the distance function defined in $\hat{P}$. Finally, in order to model the possible outcomes, the last metric space is the decision space ($\mathcal{DS}$). It is defined as $\mathcal{DS} = (O, d_O)$, where O is the set of possible outcomes and d_O is a distance function defined on O. Examples of objects for each space are presented in Table 1.1 considering two different tasks: candidate student evaluation and prediction of recidivism.

The desired outcomes of an ML model would be obtained by a function $f^* : \mathcal{CS} \to \mathcal{DS}$. The following definition of fairness that relies on f^* is then given. Let ϵ and ϵ' be fixed thresholds. Now, for any pair of objects $x, y \in P$, Firedler et al. state that f^* is (ϵ, ϵ')-fair if x and Y are close in $\mathcal{CS}$ they are also close in $\mathcal{DS}$. That is,

Table 1.1: Examples of features and outcomes on each metric space (from [32])

Construct space	**Observed space**	**Decision space**
Intelligence	IQ	Acceptance in university
Risk-adverseness	Age	Recidivism

$$d_P(x, y) \leq \epsilon \Rightarrow d_O(f(x), f(y)) \leq \epsilon'. \tag{1.1}$$

However, a direct mapping f^* from the construct space to the decision space is not possible. Instead, the actual outcomes are provided by a mapping function $f : \mathcal{OS} \to \mathcal{DS}$.

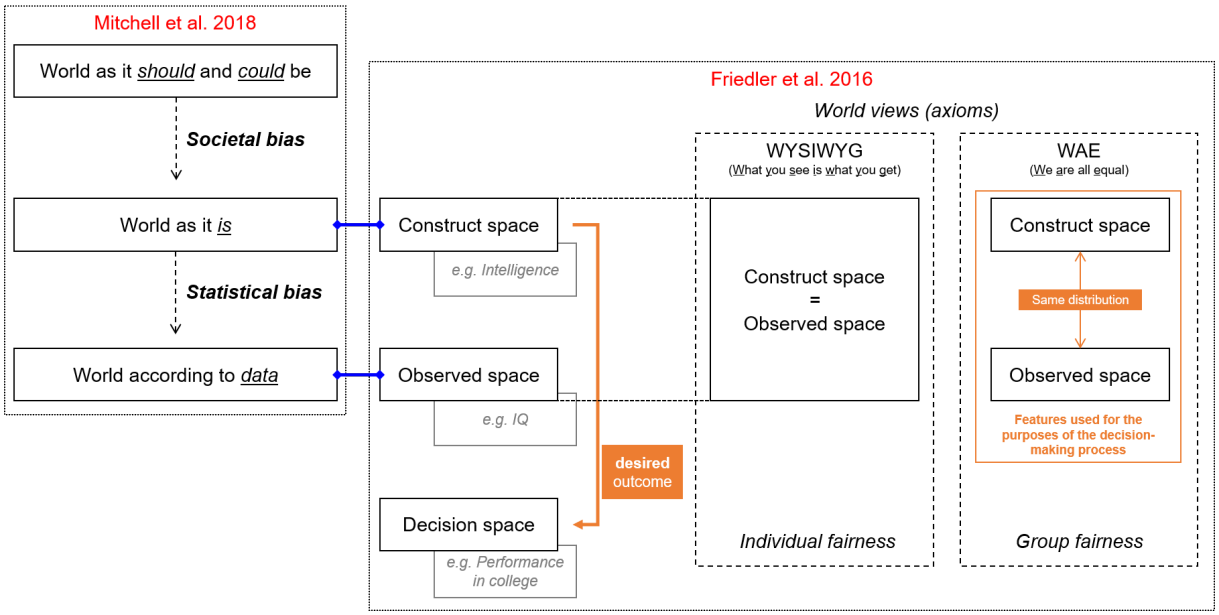


Figure 1.1: Fairness notions and its relation with the two different types of biases from [63] and the different metric spaces and world views from [32].

We now establish a relationship between the two research works from [63] and [32]. This relationship is depicted in Figure 1.1 whose some elements of both research works are connected. More precisely, the set of elements P from the construct space is equivalent to the "World as it is". Similarly, the set of elements $\hat{P}$ from the observed space is equivalent to the "World according to data". The construct and the observed spaces are considered equivalent in the WYSIWYG world view. It implies that only individual fairness is satisfy in this context. In the WAE world view, the features used in the process of decision-making are considered to have the same distribution in both spaces (construct and observed). Here, only group fairness notions can be satisfied.

1.2 Fairness notions

Fairness notions usually rely on well known scores measured with respect to privileged (*priv*) and unprivileged (*unp*) groups. For instance, with respect to "race", white people are usually the privileged group and the nonwhites the unprivileged one. Here we recall some of the best known fairness metrics.

Fairness metrics can be arranged into two categories: *group fairness* and *individual fairness*. Both categories state that sub-populations should obtain similar probabilities w.r.t. models' outcomes. They differ in the way they divide the population and the score they compute for each sub-population. On the one hand, group fairness metrics divide sub-populations based on sensitive features only (ignoring all non-sensitive features). On the other hand, individual fairness metrics take also into account non-sensitive features. Here, we recall some of the well known group fairness metrics, since group fairness is used in our proposed algorithm. We refer the reader to [60] for more details on fairness metrics.

Given a binary classification task and a model M trained on tabular data, let Y be the actual outcome, $\hat{Y}$ be the outcome returned by the model, and A^* be a particular sensitive feature. The confusion matrix is presented in Table 1.2 along with some classification metrics. These metrics will be further used to defined fairness metrics.

An example of college admission is presented in Table 1.3. In this example, Table 1.3(a) contains attributes such as GPA in high school, intelligence quotient (IQ), age and family income that can be used to evaluate whether a candidate can be admitted. The gender is also available along with the actual decision Y – if a candidate was admitted (1) of not (0). Table 1.3(b) contains the score produced by M and the outcome obtained by using 0.5 as a threshold. This example is used to illustrate the fairness notions that appear in the following.

Table 1.2: Some classification metrics based on the confusion matrix.

Total pop.		Actual outcome			
		Positive (**P**), $Y = 1$	Negative (**N**), $Y = 0$		
Predicted	Positive $\hat{Y} = 1$	**TP** (True Positive)	**FP** (False Positive)	**PPV**= $\frac{TP}{TP+FP}$ (Positive predictive value)	**FDR**= $\frac{FP}{TP+FP}$ (False discovery rate)
	Negative $\hat{Y} = 0$	**FN** (False Negative)	**TN** (True Negative)	**FOR**= $\frac{FN}{FN+TN}$ (False omission rate)	**NPV**= $\frac{TN}{FN+TN}$ (Negative predictive value)
		TPR= $\frac{TP}{TP+FN}$ (True positive rate)	**FPR**= $\frac{FP}{FP+TN}$ (False positive rate)	**OA**= $\frac{TP+TN}{TP+FP+TN+FN}$ (Overall accuracy)	**BR**= $\frac{TP+FN}{TP+FP+TN+FN}$ (Base rate prevalence)
		FNR= $\frac{FN}{TP+FN}$ (False negative rate)	**TNR**= $\frac{TN}{FP+TN}$ (True negative rate)	**BA**= $\frac{TPR+TNR}{2}$ (Balanced accurary)	

1.2.1 Group fairness

The first two fairness metrics in this chapter are based on the property of *independence*. This property implies that the classification output $\hat{Y}$ is statistically independent of the sensitive feature A^*, i.e., $\hat{Y} \perp A^*$ (or $S \perp A^*$).

- **Demographic Parity (DP)** [44] is defined as the difference in the predicted acceptance rates between the unprivileged and privileged groups. That is, DP requires that $(TP + FP)/(TP + FP + TN + FN)$ must be the same for both sub-populations.

$$\mathbb{P}[\hat{Y} \mid A^* = unp] = \mathbb{P}[\hat{Y} \mid A^* = priv]$$

  Example. In the college admission example from Table 1.3, the predicted acceptance rates for male and female candidates are 0.57 and 0.4 respectively. It means that DP does not hold.

Table 1.3: A simple college admission example. Y represents the data label indicating if the candidate is admitted (1) or not (0). $\hat{Y}$ is the prediction which is based on the score S (a threshold of 0.5 is used).

(a) Dataset

Gender	IQ	GPA	Age	Family Income	Y
Female 1	84	3.3	18	low	1
Female 2	84	3.3	27	high	0
Female 3	140	4.7	22	low	1
Female 4	113	4.9	20	low	1
Female 5	120	3.9	24	high	0
Male 1	84	2.0	33	low	0
Male 2	120	3.9	24	high	1
Male 3	55	3.9	30	high	1
Male 4	84	3.3	28	high	0
Male 5	99	3.3	19	low	1
Male 6	84	3.3	18	low	1
Male 7	140	4.8	21	high	1

(b) Prediction

$\hat{Y}$	S
0	0.1
1	0.5
1	0.5
0	0.3
0	0.2
0	0.1
1	0.8
0	0.1
1	0.5
0	0.3
1	0.5
1	0.8

Table 1.4: Classification metrics for the example of college admission (**Male**)

Total pop.		Actual outcome			
		$Y=1$	$Y=0$		
Pred	$\hat{Y}=1$	**TP**= 3	**FP**= 1	**PPV**= 0.75	**FDR**= 0.25
	$\hat{Y}=0$	**FN**= 2	**TN**= 1	**FOR**= 0.67	**NPV**= 0.33
		TPR= 0.60	**FPR**= 0.50	**OA**= 0.57	**BR**= 0.71
		FNR= 0.40	**TNR**= 0.50	**BA**= 0.45	

Table 1.5: Classification metrics for the example of college admission (**Female**)

Total pop.		Actual outcome			
		$Y=1$	$Y=0$		
Pred	$\hat{Y}=1$	**TP**= 1	**FP**= 1	**PPV**= 0.50	**FDR**= 0.50
	$\hat{Y}=0$	**FN**= 2	**TN**= 1	**FOR**= 0.67	**NPV**= 0.33
		TPR= 0.33	**FPR**= 0.50	**OA**= 0.40	**BR**= 0.60
		FNR= 0.67	**TNR**= 0.50	**BA**= 0.58	

- **Disparate Impact (DI)** [28] is rooted in the desire for different groups to experience similar rates of positive decision outcomes ($\hat{Y}=1$).

$$DI = \frac{\mathbb{P}[\hat{Y}=1|A^*=unp]}{\mathbb{P}[\hat{Y}=1|A^*=priv]}$$

  Example. Similarly to DP, DI does not hold for the example as the proportion of positive outcomes is $0.4/0.57 = 0.7$ (which is different from 1, the optimal value).

The following fairness notions are formulated based on the property of *separation*. This property implies that the classification output $\hat{Y}$ is conditionally independent of the sensitive feature A^* given the actual classification Y, i.e., $\hat{Y} \perp A \mid Y$ (or $S \perp A \mid Y$).

- **Equalized Odds (EqOdds)** holds if true positive rates (**TPR**) and false positive rates (**FPR**) are the same for both sub-populations (unprivileged and privileged groups).

$$\mathbb{P}[\hat{Y} = 1 \mid Y = y, A^* = unp] = \mathbb{P}[\hat{Y} = 1 \mid Y = y, A^* = priv], \forall y \in [0, 1]$$

Example. Since both TPR and FPR must be the same for the two sub-populations, EqOdds does not hold in the college admission example. Even though the FPR is 0.50 for both groups, the TPR is 0.60 and 0.33 for male and female groups respectively.

  - **Equal Opportunity (EO)** (or *disparate mistreatment*) [87] is the difference in true positive rates (**TPR**) between the unprivileged and privileged groups ($TPR_{unp} = TPR_{priv}$). It is a relaxation of Equalized Odds, since EO is insensitive to FPR.

$$\mathbb{P}[\hat{Y} = 1 \mid Y = 1, A^* = unp] = \mathbb{P}[\hat{Y} = 1 \mid Y = 1, A^* = priv]$$

Example. Like EqOdds, EO does not hold for the example from Table 1.3, since the TPR is different for male and female sub-populations.

  - **Predictive Equality (PE)** [60] is computed as the difference in false positive rates (**FPR**) between unprivileged and privileged groups ($FPR_{unp} = FPR_{priv}$). It is a relaxation of Equalized Odds, since PE is insensitive to TPR.

$$\mathbb{P}[\hat{Y} = 1 \mid Y = 0, A^* = unp] = \mathbb{P}[\hat{Y} = 0 \mid Y = 1, A^* = priv]$$

Example. Since there exists FPR equality between male and female groups and PE is not sensitive to TPR, PE holds in the example of college admission.

- **Balance for Positive Class (BP)** relies on the score instead of the derived outcome. This notion holds if the average score for the positive class is equal for unprivileged and privileged groups.

$$\mathbb{E}[S \mid Y = 1, A^* = unp] = \mathbb{E}[S \mid Y = 1, A^* = priv]$$

Example. In the college admission example from Table 1.3, the calculated average scores for the positive class are

$$\mathbb{E}[S \mid Y = 1, A^* = \text{“female”}] = 0.3,$$
$$\mathbb{E}[S \mid Y = 1, A^* = \text{“male”}] = 0.5.$$

Based on these values, BP does not hold as the expected score values are 0.5 and 0.3.

- **Balance for Negative Class (BN)** is similar to the Balance for Positive Class, but reliant on the scores for negative class for all groups.

$$\mathbb{E}[S \mid Y = 0, A^* = unp] = \mathbb{E}[S \mid Y = 0, A^* = priv]$$

Example. Like BP, we calculate the average scores for the negative class in order to evaluate BN. We thus obtain the following average scores,

$$\mathbb{E}[S \mid Y = 0, A^* = \text{“female”}] = 0.35,$$

$$\mathbb{E}[S \mid Y = 0, A^* = \text{“male”}] = 0.3.$$

Since the expected scores for the negative class are different, BN does not hold either.

The third property is *sufficiency*. It implies that both sensitive feature A^* and actual outcome Y are conditionally independent given the classification output $\hat{Y}$, i.e., $Y \perp A^* \mid \hat{Y}$ (or $Y \perp A^* \mid S$). The following fairness metrics take this property into account.

- **Conditional use accuracy equality (CAE)** holds if both groups have equal **PPV** and **NPV**.

$$\mathbb{P}[Y = 1 \mid \hat{Y} = y, A^* = unp] = \mathbb{P}[Y = 1 \mid \hat{Y} = y, A^* = priv]$$

Example. In the college admission example, the PPVs for male and female are 0.75 and 0.50 respectively. The NPVs for male and female are equal. Since PPV and NPV must be equal for both groups, in this example, CAE does not hold.

  - **Predictive Parity (PP)** is achieved when the difference in **PPV** between unprivileged and privileged groups is zero.

$$\mathbb{P}[Y = 1 \mid \hat{Y} = 1, A^* = unp] = \mathbb{P}[Y = 1 \mid \hat{Y} = 1, A^* = priv]$$

Example. Since PP is insensitive to NPV, PP holds (PPV is equal for male and female groups) in the college admission example.

- **Calibration** holds if, for each predicted probability score $S = s$, both groups have the same probability to belong to the positive class

$$\mathbb{P}[Y = 1 \mid S = s, A^* = unp] = \mathbb{P}[Y = 1 \mid S = s, A^* = priv], \forall s \in [0, 1]$$

Example. In the college admission example, we must take into account all scores to calculate calibration. That is, based on Table 1.3 we calculate the probability to belong to the positive class for each predicted probability score in $S = \{0.1, 0.3, 0.5, 0.8\}$. The calculated probabilities to actually belong to the positive class are presented in Table 1.6. Since the probabilities are different between the two sub-populations for $\{0.1, 0.3, 0.8\}$, calibration does not hold in this example.

Table 1.6: Calibration for the college admission example.

s	0.1	0.3	0.5	0.8
Female	0	0	1	0
Male	0	0	1	1

- **Equal Accuracy (EA)** [44] is defined as the difference in accuracy score (**OA**) between unprivileged and privileged groups ($OA_{unp} = OA_{priv}$).

$$\mathbb{P}[\hat{Y} = Y \mid A^* = unp] = \mathbb{P}[\hat{Y} = Y \mid A^* = priv]$$

Example. In the college admission example, the calculated overall accuracy for male and female candidates are 0.57 and 0.40 respectively. Hence, EA does not hold for this example.

Table 1.7: Group fairness metrics and their equalizing targets, insensitivity, and properties

Property	Notion	Equalizing	Insensitive
Independence	DP, DI	Positive outcome	Negative outcome
Separation	EqOdds	TPR, FPR	
	EO	TPR	FPR
	PE	FPR	TPR
	BP	Avg prob positive	Avg prob negative
	BN	Avg prob negative	Avg prob positive
Sufficiency	CAE	PPV, NPV	
	PP	PPV	NPV
	Calibration		

A summary of the fairness metrics presented here are presented in Table 1.7 with their equalizing targets, properties and the accepted inequalities or insensitiveness. We now present some equivalences between fairness notions.

- Satisfying both predictive equality (PE) and equal opportunity (EO) is equivalent to satisfying equalized odds (EqOdds).

$$EqOdds \Leftrightarrow PE \wedge EO$$

- Satisfying equalized odds (EqOdds) and conditional use accuracy equality (CAE) leads to also satisfying overall accuracy (OA).

$$EqOdds \wedge CAE \Rightarrow OA$$

1.2.2 Individual fairness

Although group fairness notions can be suitable for policies, they can hide unfairness issues since they ignore non-sensitive features. Individual fairness instead takes non-sensitive features into account. We present some of them next.

- **Fairness through awareness (FTA)** [28] states that for each pair of data instances (e.g. two different individuals) x_i and x_j, the difference between probability distributions of $\mathcal{P}(x_i)$ and $\mathcal{P}(x_i)$ should not be greater than the distance between these two individuals. That is, similar individuals should receive similar outcomes from a MLDM system. It can be formally expressed as

$$\delta(\mathcal{P}(x_i), \mathcal{P}(x_j)) \leq d(x_i, x_j),$$

  where $d(\cdot, \cdot)$ is a distance function defined on the set of instances and $\delta(\cdot, \cdot)$ is a distance function defined on the probability distribution. One of the difficulties in using this notion is that the distance function should be known beforehand.

- **Fairness through unawareness**[9] **(FTU)** [63] is a simple and straightforward approach to address fairness issues where we ignore completely any sensitive feature while training

[9] Also know as anti-classification [21] and treatment parity [58].

the ML model. In other words, a model is fair if it does not use any sensitive features to produce the outcomes. That is, any model that completely excludes the set of sensitive features satisfy this notion.

1.3 Process fairness

Process fairness [37] can be described as a set of subjective fairness notions that are centered on the process that leads to outcomes. These notions are not focused on the fairness of the outcomes, instead they quantify the fraction of users that consider fair the use of a particular set of features. They are subjective as they depend on user judgments which may be obtained by subjective reasoning.

For instance, in recidivism risk prediction, a user may consider the use of a particular feature unfair, but if the use of the same feature increases the accuracy of a model, the same user may consider it fair. Since people may change their judgments over time or in different contexts, Grgić-Hlača et al. [37] propose three different measures of process fairness to capture this subjective reasoning, namely: feature-apriori fairness, feature-accuracy fairness, and feature-disparity fairness.

Given a model M, we denote $M_{\mathcal{F}}$ the model that has been trained using the set of features $\mathcal{F}$. Now, let $\mathcal{U}$ be the set of all users (or members of society). Measures of process fairness rely on the judgment of $\mathcal{U}$ on $M_{\mathcal{F}}$ in the following way.

- **Feature-apriori fairness.** It measures the proportion of users that consider fair the use of a particular feature A without any prior knowledge of the effect of the employment of A. We denote $\mathcal{U}_f \subseteq \mathcal{U}$ the subset of users that consider the use of A fair in this context. For a given subset of features $\mathcal{F}' \subseteq \mathcal{F}$, feature-apriori fairness of $M_{\mathcal{F}'}$ is calculated as

$$\frac{|\bigcap_{A_i \in \mathcal{F}'} \mathcal{U}_{A_i}|}{|\mathcal{U}|}.$$

- **Feature-accuracy fairness.** It measures the proportion of users that consider fair the use of A if the employment of A increases the accuracy. More precisely, given the accuracy of a model $Acc(\cdot)$, we denote $\mathcal{U}_f^{Acc} \subseteq \mathcal{U}$ the subset of users that consider the use of A fair if it increases the accuracy $Acc(M)$. We thus compute the feature-accuracy fairness of $M_{\mathcal{F}'}$ as

$$\frac{|\bigcap_{A_i \in \mathcal{F}'} Condition(\mathcal{U}_{A_i}, \mathcal{U}_{A_i}^{Acc})|}{|\mathcal{U}|},$$

  where

$$Condition(\mathcal{U}_{A_i}, \mathcal{U}_{A_i}^{Acc}) = \begin{cases} \mathcal{U}_{A_i} \cup \mathcal{U}_{A_i}^{Acc}, & \text{if } Acc(M_{\mathcal{F}'}) > Acc(M_{\mathcal{F}' \setminus \{A_i\}}) \\ \mathcal{U}_{A_i}, & \text{if } Acc(M_{\mathcal{F}'}) \leq Acc(M_{\mathcal{F}' \setminus \{A_i\}}). \end{cases}$$

- **Feature-disparity fairness.** It measures the proportion of users that consider fair the use of a feature A if the feature helps to increase disparity. More precisely, given a measure of disparity $Disp(\cdot)$, we denote $\mathcal{U}_f^{Disp} \subseteq \mathcal{U}$ the subset of users that consider the use of

A fair if the presence of A increases the disparity of the model. We thus compute the feature-disparity fairness of $M_{\mathcal{F}'}$ as

$$\frac{|\bigcap_{A_i \in \mathcal{F}'} Condition(\mathcal{U}_{A_i}, \mathcal{U}_{A_i}^{Disp})|}{|\mathcal{U}|},$$

where

$$Condition(\mathcal{U}_{A_i}, \mathcal{U}_{A_i}^{Disp}) = \begin{cases} \mathcal{U}_{A_i} \cup \mathcal{U}_{A_i}^{Disp}, & \text{if } Disp(M_{\mathcal{F}'}) \leq Disp(M_{\mathcal{F}' \setminus \{A_i\}}) \\ \mathcal{U}_{A_i}, & \text{if } Disp(M_{\mathcal{F}'}) > Disp(M_{\mathcal{F}' \setminus \{A_i\}}). \end{cases}$$

Example. An estimation of these measures of process fairness w.r.t. some features from the Compas dataset is presented in [37]. The Compas dataset contains information compiled by ProPublica [7] and the goal is to predict the two-year violent recidivism. That is, whether a convicted individual would commit a violent crime in the following two years or not.

In [37], six features were selected and 100 users were asked to answer a survey in which they basically should tell if the employment of some particular feature would be fair or not. More precisely, three main questions were included in a survey. Each question was used to estimate one of the three measures of process fairness: feature-apriori fairness (f-apriori), feature-accuracy fairness (f-accuracy), and feature-disparity fairness (f-disparity). Table 1.8 contains the estimations for the three measures. Here, we focus only on three features ("age", "gender", and "race"), even though in the original study six features were taken into consideration. Note that the use of "race" is judged the most unfair among the other features.

Table 1.8: The calculated measures of process fairness based on user judgments in the case of classifiers trained on the Compas dataset. Example taken from [37].

	f-apriori	**f-accuracy**	**f-disparity**
age	0.44	0.61	0.32
gender	0.26	0.55	0.24
race	0.21	0.42	0.17

In the original study [37], Grgić-Hlača et al. also compare the calculated measures of process fairness along with classification accuracy. For instance, they took some trained classifiers – such as the one that presents the highest accuracy and the most fair classifier – and then analyze the classification accuracy along with the calculated measures of process fairness. They noticed that the classifier that obtained the highest accuracy had low values for the measures of process fairness. On the other hand, the classifier with the highest values w.r.t process fairness obtained lower accuracy.

1.3.1 Improving process fairness

A natural approach to improve process fairness is to remove all sensitive (protected or salient) features before training classifiers. This simple approach connects process fairness to fairness through unawareness. However, dropping out sensitive features may impact negatively classification performance [37, 87]. Addressing the tension between these two constraints– classification performance and process fairness– requires to explore the set of classifiers that have suitable classification performance and at the same time low dependence on sensitive features.

A Rashomon set [30] is defined as a set of ML models that present similar performances in terms of error rate (the "good models") but that utilize features differently, e.g., they rely on class labels or certain features at different levels. Breiman [16] used "Rashomon effect" to denote a multiple functions with similar error rates but different descriptions. Recently, Coston et al. [22] adapted the notion of Rashomon set by integrating fairness metrics. In this thesis, we are interested in classifiers that belong to the set of "good" models, i.e. they have similar classification performance, but are less reliant on sensitive features. In order to quantify classifiers' reliance on sensitive features, we take advantage of explanation methods, which is discussed in Chapter 3.

1.4 Tensions between fairness notions

Under the assumptions of independence ($\hat{Y} \perp A^*$), separation ($\hat{Y} \perp A^* \mid Y$) and sufficiency ($Y \perp A^* \mid \hat{Y}$), one can identify tensions and impossibilities between fairness notions. It has been shown that there are incompatibilities (or impossibilities) among fairness notions [8, 63]. These impossibilities indicate that it is not always possible to satisfy some fairness notions at the same time. A deeper discussion about tensions and impossibilities between fairness notions can be found in [8, 63].

In the presence of incompatibilities between fairness notions, a trade-off should be considered when designing (or learning) a MLDM system. More precisely, two solutions are possible: (1) making a choice of which notion must be satisfied or (2) considering a relaxation of fairness notions instead of their strict definitions. In the first solution, one fairness notion is left out and that is not satisfied. In the case of the second solution, relaxations must be used instead of strict notions of fairness. That is applied for separation and sufficiency since there are relaxed notions for these two properties.

We now briefly present some incompatibilities and give examples of how to tackle them. We state that a model is useless when its predictions $\hat{Y}$ and Y are independent ($\hat{Y} \perp Y$). Also, we state that base rates (BR) are the proportion of positive outcomes in a population (Table 1.2).

- **Demographic parity *versus* conditional use accuracy equality**

  Demographic parity is based on the property of independence while conditional use accuracy equality is based on sufficiency. Independence and (strict) sufficiency are incompatible unless base rates of the privileged and unprivileged groups are equal or the model is useless.

$$\underbrace{\hat{Y} \perp A^*}_{\text{(independence)}} \quad \wedge \quad \underbrace{Y \perp A^* \mid \hat{Y}}_{\text{(strict sufficiency)}} \quad \Longrightarrow \quad \underbrace{Y \perp A^*}_{\text{(equal base rates)}} \quad \vee \quad \underbrace{\hat{Y} \perp Y}_{\text{(useless model)}}$$

  As mentioned previously, there are relaxations of sufficiency. So, one approach to break this impossibility is by replacing conditional use accuracy equality by predictive parity instead.

- **Demographic parity *versus* equalized odds**

  Equalized odds is based on separation, which is also incompatible with independence (demographic parity). Similarly to the previous incompatibility, this one also holds if both groups (privileged and unprivileged) have equal base rates or the model provides useless predictions.

$$\underbrace{\hat{Y} \perp A^*}_{\text{(independence)}} \quad \wedge \quad \underbrace{\hat{Y} \perp A^* \mid Y}_{\text{(strict separation)}} \quad \Longrightarrow \quad \underbrace{Y \perp A^*}_{\text{(equal base rates)}} \quad \vee \quad \underbrace{\hat{Y} \perp Y}_{\text{(useless model)}}$$

One way to overcome this incompatibility is to consider a relaxation of separation. That is, instead of taking into account equalized odds, one use either equality of opportunity or predictive equality in order to break the incompatibility.

- **Equalized odds *versus* conditional use accuracy equality**

  Strict separation and strict sufficiency are also incompatible and so are equalized odds and conditional use accuracy equality. Again, it does not hold when base rates for subgroups are equal.

$$\underset{\text{(strict separation)}}{\hat{Y} \perp A^* \mid Y} \quad \wedge \quad \underset{\text{(strict sufficiency)}}{Y \perp A^* \mid \hat{Y}} \quad \Rightarrow \quad \underset{\text{(equal base rates)}}{Y \perp A^*}$$

  In the presence of this incompatibility, we should consider relaxations of separation and sufficiency. For instance, predictive equality instead of equalized odds and predictive parity rather than conditional use accuracy equality.

Chapter 2

Unfairness mitigation

Contents

2.1 Pre-processors 24
2.1.1 Reweighing [17] 24
2.1.2 FairExp [75] 24
2.1.3 FairBatch [74] 25
2.2 In-processors 25
2.2.1 Debiasing with adversarial learning [91] 25
2.2.2 Exponentiated gradient and grid search reduction [2] 25
2.2.3 Prejudice remover [50] 26
2.2.4 AdaFair [46] 26
2.3 Post-processors 26
2.3.1 Threshold Optimizer [44] 27
2.3.2 Reject option classification [49] 27
2.3.3 Shifted Decision Boundary (SBD) [29] 27
2.4 Hybrid processors 27
2.4.1 Learning Fair Representations (LRF) [90] 27
2.4.2 Fairness-Aware Ensemble (FAE) [45] 28
2.5 The fairness-accuracy trade-off 28

Fairness processors are algorithmic approaches (also know as algorithmic interventions and fairness-enhancing interventions) that are conceived to optimize one or more fairness notions. We give an overview of algorithmic interventions in this chapter. These approaches are often arranged based on the stage they apply fairness interventions in a ML pipeline (see Figure 2.1), namely: *pre-processing* (Section 2.1), *in-processing* (Section 2.2), and *post-processing* (Section 2.3).

Particularly, we focus on methods whose implementations (source codes) are available and that covers all categories. A more complete list of fairness processors can be found in [33, 68, 71].

In this thesis, we propose the inclusion of a fourth category named as *hybrid-processing*, which comprises algorithmic approaches that combine different fairness interventions as a single method and, as a consequence, do not fit in any of the three traditional categories (Section 2.4).

Table 2.1 presents a summary of the fairness-enhancing interventions described here. It also indicates the links for the code artifacts (git repositories or the Python packages for fairness: AIF 360[10] or Fairlearn[11]).

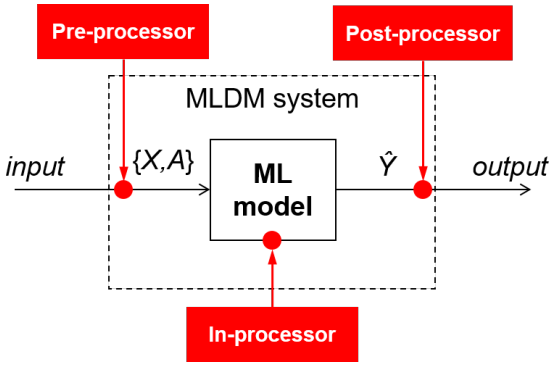


Figure 2.1: Stages for fairness intervention.

2.1 Pre-processors

Pre-processing approaches (pre-processors) modify the input in order to achieve fair outcomes. These processors can be applied to any model, since they are model-agnostic. However, the fact they change the input before training may harm the explainability. In addition, pre-processors may increase the uncertainty of the classification process which impacts the level of accuracy [71].

2.1.1 Reweighing [17]

This processor assigns different weights to data instances based on the distribution of a sensitive feature and the class label. The weights are used to guide a sampling procedure (with replacement) in order to obtain a (new) balanced dataset whose sensitive feature and class label are independent. For instance, data instances that obtained high weights will reappear more often in the balanced dataset. A classifier is then trained on the balanced dataset. As a consequence of the sampling procedure, classification errors on the high weighted data instances are more expensive.

$$w(Y = 1, A^* = unp) = \frac{\mathbb{P}_{exp}[Y = 1, A^* = unp]}{\mathbb{P}_{act}[Y = 1, A^* = unp]}$$

2.1.2 FairExp [75]

FairExp is a pre-processor that uses feature engineering to discover a set of features in order to achieve simultaneously high accuracy and fairness. Given a biased dataset, this processor obtains a (new) derived dataset that is less biased than the original one. First, it starts by discovering a set of constructed features that increases accuracy. This process is guided by the description length (from *Minimum Description Length*, MDL) of the obtained features. It then reduces the set of features by selecting a subset that presents high-accuracy and simultaneously improve fairness. FairExp relies on the causal notion of fairness *ratio of observational discrimination* [76]

[10] https://github.com/Trusted-AI/AIF360
[11] https://github.com/fairlearn/fairlearn

that does not require the causal graph to be computed. The main idea is to extract unbiased information from biased features, and then include it into the set of constructed features.

2.1.3 FairBatch [74]

This pre-processor is an extension of a batch selection algorithm that modifies the batch training in order to enforce model fairness (e.g., equal opportunity, equalized odds, demographic parity). More precisely, it measures fairness and adapt the size of the batch based on sensitive groups (which links this pre-processor with the reweighing approach).

2.2 In-processors

In-processing techniques (in-processors) try to change the learning algorithm during the training process. Since they are an easy way to impose fairness constraints, these processors usually take into account the tension between fairness and classification performance. However, they can not always be applied to any model since they are usually model-specific.

2.2.1 Debiasing with adversarial learning [91]

This in-processor trains two neural networks: (1) a predictor and (2) an adversary. Both networks have different objectives since the goal is, at the end of the training process, to attain the separation criterion. The goal of the predictor is to learn a function that predicts the class label (or the score in a regression problem), while the adversary takes as input the predicted label and its goal is to predict a sensitive feature. The predictor has weights W and loss function L_P, while the adversary has weights U and loss function L_A.

The main idea behind this processor comes in the way the weights of both networks are updated. The weights U (of the adversary) are modified based only on the gradient $\nabla_U L_A$. Unlike the adversary's weights, the update of the predictor's weights, W, relies on two components: the first one is the gradient $\nabla_W L_P$ (that minimizes predictor's loss function), and the second one is $\text{proj}_{\nabla_W L_A} \nabla_W L_P - \alpha \nabla_W L_A$ that avoids the predictor from helping or not trying to harm the adversary.

This algorithmic intervention can be applied on classification and regression problems. Also, it can improve demographic parity, equalized odds, and equality of opportunity.

2.2.2 Exponentiated gradient and grid search reduction [2]

These in-processors reduce an unfairness mitigation problem to a sequence of cost-sensitive classification problem. Here, fairness notions are re-written in a generic constraint format (a.k.a. a vector of conditional moments), so that processors can support multiple fairness notions. More precisely, the statistical parity notion is a special case of the generic constraint that follows

$$\mu_j(h) = \mathbb{E}[g_j(X, A, Y, h(X)) \mid \varepsilon_j],$$

where g_j is a function that maps a data instance (sensitive and non-sensitive features) along with the predicted and actual outcomes into [0,1], h is a classifier, and ε_j is an event, which is independent from h, that relies on features (sensitive and non-sensitive) and Y. The idea is then to solve the following problem

$$M\mu(h) \leq c, \tag{2.1}$$

where M is a matrix $\mathcal{K} \times \mathcal{J}$, and c is a vector describing linear constraints. In order to empirically solve Eq. 2.1, $\mu(h)$ and c are replaced by $\hat{\mu}(h)$ and $\hat{c}$, respectively, in the form of a Lagrangian

$$L(h, \lambda) = \mathbb{P}[h(X) = Y] + \lambda^T (M\hat{\mu}(h) - \hat{c}),$$

where $\lambda \in \mathbb{R}_+^{|\mathcal{K}|}$ is a Lagrange multiplier. Now, one needs to find a saddle point that is the overlapped point between maximizing L and minimizing L. After few iterations of updating λ, the optimal h is obtained as a result.

2.2.3 Prejudice remover [50]

This in-processor relies on the prejudice index (PI) that measures the indirect prejudice. PI quantifies the mutual information between class label and a sensitive feature, which indicates the degree of dependence on sensitive information.

$$PI = \sum_{(Y,A) \in D} [Y, A] \ln \left(\frac{[Y, A]}{[Y][A]} \right)$$

PI is then included as a regularizer in the optimization function (see below), to take fairness into account. A parameter η is used to (reduce) enforce the importance of (un)fairness in the training process. The idea behind the penalty (along with the parameter η) is to reduce the dependency of the model on sensitive information and its fairness.

$$\min_f L[f(x), Y] + \eta PI$$

Prejudice remover can be applied to any discriminative probabilistic classifier. The original paper [50] employed this in-processor in a logistic regression model.

2.2.4 AdaFair [46]

It is an ensemble approach that trains a set of classifiers (weak learners) and then combines the output of these classifiers as a single outcome. More precisely, it is an extension of AdaBoost that takes into consideration Equalized Odds as a fairness notion to be optimized during training. In order to do that, it uses the notion of *cumulative fairness* which take into account the fairness of a weak learner in the past iterations, instead of only considering the current iteration. In addition, AdaFair uses confidence scores in the re-weighting process in order to obtain fairer model's outcomes, which differs from the traditional AdaBoost that relies only on classification error.

2.3 Post-processors

Post-processing approaches (post-processors) modify algorithm's outputs to satisfy fairness constraints. They are usually model-agnostic and also agnostic to the input data, which make them easier to implement. However, post-processors usually present low performance compared to pre-processors [54, 71].

There exist two categories of post-processors: (1) transparent-box approaches that change the decision boundary of a model , and (2) opaque-box approaches that directly change the classification label.

2.3.1 Threshold Optimizer [44]

It finds a threshold τ for each group (w.r.t. a sensitive feature) that minimizes a loss function L and, at the same time, takes into consideration the separation criterion (either equalized odds or equal opportunity). This post-processor can be employed on top of any model and it does not require the information about non-sensitive features X, as it is based only on the joint distribution of the sensitive feature A, and the predicted and actual outcomes, $\hat{Y}$ and Y respectively. The optimization process searches in the receiver operating characteristic (ROC) curves between true and false positive rates of sub-populations. More precisely, for each binary sensitive feature, this processor finds an intersection point between the ROC curves of the two sub-populations, since the shape of the curves and the cost of misclassification are not necessarily the same for different groups. In other words, it minimizes the loss of classification while improves equalized odds (or equal opportunity) as described as follows

$$\min_{\tau}[S(X|A=a,Y=0) \leq \tau]\cdot L(\hat{Y}=1,Y=0)+(1-[S(X|A=a,Y=1) > \tau])\cdot L(\hat{Y}=0,Y=1).$$

2.3.2 Reject option classification [49]

This post-processor requires that classifiers output probability scores. Data instances that obtain scores close to 0 or 1 indicate that a classifier has high confidence (low uncertainty) about predicting the class labels of those instances. Instead, Reject option classification focuses on the other group of data instances. It relies on the idea that data instances that have high uncertainty can have their class labels switched in order to enforce fairness. Once the definition of high uncertainty (critical region) is established by the user, this processor changes label of instances in the critical region to improve a certain fairness notion.

2.3.3 Shifted Decision Boundary (SBD) [29]

This post-processor is also inspired by the boosting mechanism. Unlike the traditional AdaBoost classifier that applies a majority voting to obtain the outcome, SBD uses a confidence score (and not a classification error) for aggregating and obtaining the class label. Here, the statistical parity notion is incorporated into the confidence score, which is originally defined only by the distance of a data instance to the decision boundary. As a result, this strategy moves the boundary decision towards fairer outcomes (w.r.t. statistical parity). SBD can be employed on top of any model but it also allows the use of different fairness metrics.

2.4 Hybrid processors

These processors combine more than one algorithmic intervention in an ML pipeline. The idea is to use the advantages of a fairness processor to overcome the disadvantages of another processor.

2.4.1 Learning Fair Representations (LRF) [90]

This fairness processor transforms (encodes) the data from an original space to a representation space in order to meet the following requirements: (1) to optimize statistical parity, (2) to lose the membership information about protected groups while keeping any other information from the original space, and (3) to map data instances from the representation space to Y such that the mapping has similar performance compared to an optimal classifier, i.e., to keep the highest

Table 2.1: Summary of some fairness-enhancing interventions.

Type	Method	Fairness notion	Type of model	Artifact availability
Pre-proc.	Reweighing [17]	Statistical parity	Model agnostic	AIF 360
	FairExp [75]	Ratio of observational discrimination	Model agnostic	Git repository[12]
	FairBatch [74]	Equalized odds, Statistical parity	Model agnostic	-
	Debiasing with adversarial learning [91]	Separation criterion	Gradient-based	AIF 360
In-proc.	Exponentiated Gradient [2, 3]	Equalized odds, Statistical parity	Model agnostic	AIF 360, Fairlearn
	Prejudice Remover [50]	Normalized prejudice index	Logistic regression	AIF 360
	AdaFair [46]	Equalized odds	AdaBoost	Git repository[13]
Post-proc.	Threshold Optimizer [44]	Equalized odds	Any score based	AIF 360, Fairlearn
	Reject option classification [49]	Independence criterion	Model agnostic	AIF 360
	SBD [29]	Statistical parity	Any score based	Git repository[14]
Hybrid	LRF [90]	Statistical parity	Logistic regression	AIF 360
	FAE [45]	Equal opportunity	Bagging and boosting based	Git repository[15]
	FixOut [5, 12]	Process fairness	Any score based	Git repository[16]

possible accuracy. In order to do so, LRF takes into account these requirements by solving an objective function thanks to an in-processing approach. The goal is then to minimize the loss of a multi-objective and non-linear function.

2.4.2 Fairness-Aware Ensemble (FAE) [45]

This is a framework for fairness-aware classification that combines two fairness-enhancing interventions: pre-processing and post-processing. The first one tackles the problem of group imbalance and class imbalance by generating samples before the training phase (pre-processing intervention). The second one moves the decision boundary towards fairer outcomes (post-processing intervention) based on the fairness notion Equal Opportunity.

2.5 The fairness-accuracy trade-off

This section focuses on the tension between fairness and classification accuracy, which is also known as the fairness-accuracy trade-off [53]. This tension naturally arises in many real-world scenarios, e.g., mortgage lending [55]. It is discussed in several papers [9, 33, 62, 92] and it arises once we try to improve fairness in a ML pipeline by using a fairness processor.

Even though various fairness processors take into account both fairness and classification accuracy during the fairness intervention, there is still room for studying, characterizing and

defining this trade-off. On the one hand, distinct conclusions have been found about the impact on the classification accuracy when fairness is enforced. For instance, one can say that improving fairness can compromise accuracy [53], however, in specific contexts, it can actually increase accuracy [70, 86].

On the other hand, other papers are focused on characterizing or questioning the underlying assumptions made in previously published studies. For instance, [92] shows that in the evaluation of the fairness-accuracy trade-off, the acceptance rate must be taken into account, since classification accuracy from distinct acceptance rates can not be comparable. More precisely, they rely on a notion of discrimination to assess fairness and show that better classification accuracy does not necessarily mean better classification if it comes from distinct acceptance rates. More recently, [20] argue that researchers make assumptions that may lead to actually unfairness outcomes (or emergent unfairness). More precisely, three unsuitable assumptions are indicated: (1) fairness metrics are sufficient to assess fairness, (2) the lack of consideration of historical context, and (3) collecting more data on protected groups as an adequate solution.

[13] https://github.com/iosifidisvasileios/AdaFair
[14] https://github.com/j2kun/fkl-SDM16
[15] https://github.com/iosifidisvasileios/Fairness-Aware-Ensemble-Framework
[16] https://gitlab.inria.fr/galvesda/fixout

Chapter 3

Explanation methods

Contents

3.1 Explanation methods: properties and taxonomies 31
3.2 Local post-hoc explanations 33
3.2.1 Local Interpretable Model Agnostic Explanations (LIME) [72] . . . 34
3.2.2 SHapley Additive exPlanations (SHAP) [59] 36
3.3 From local to global explanations 37
3.4 Drawbacks and pitfalls of employing LIME and SHAP 38
3.5 LIME and SHAP extensions 39
3.6 PathExplain [47] 40

This chapter is focused on methods that generate explanations for ML models. We start by presenting some taxonomies of explanations methods in Section 3.1. Then, in Section 3.2, we focus on specific explanation methods. More precisely, two methods that produce explanations for individual predictions are detailed since these methods are needed later on to understand the contributions of this thesis. Next, in Section 3.3, we present some approaches to obtain global explanations from local explanations. In Section 3.4, we point out some drawbacks of employing these particular explanation methods. Finally, we end this chapter in Section 3.5 by briefly mention some new extensions of those methods.

3.1 Explanation methods: properties and taxonomies

The advent of opaque, high-stack ML models along with the new regulations (e.g. GDPR) have motivated a new body of study dedicated to explainability [41]. Since explainability in ML has became a large field, in this thesis, we focus on the problem of explainability of black box models.

We adopt the notion of black box models from [41]. That is, these models are known to be either non-transparent (i.e. opaque) or transparent but uninterpretable to the observers (e.g. users, practitioners). We thus consider, for instance, Neural Networks, Tree Ensembles, and Support Vector Machines as black box models, while Decision Trees, Decision Rules and Linear Regressions are considered as transparent models.

The opaque nature of black box models impose challenges of explanability. That is, there is a need to open the black boxes in order to understand and explain their inner workings and outcomes. This problem is called as *black box explanation problem*, and it can be separated in three sub-problems, namely: *outcome explanation*, *model explanation*, and *model inspection*.

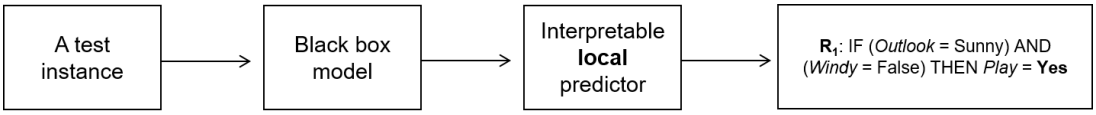


Figure 3.1: An example of the outcome explanation problem (from [41]).

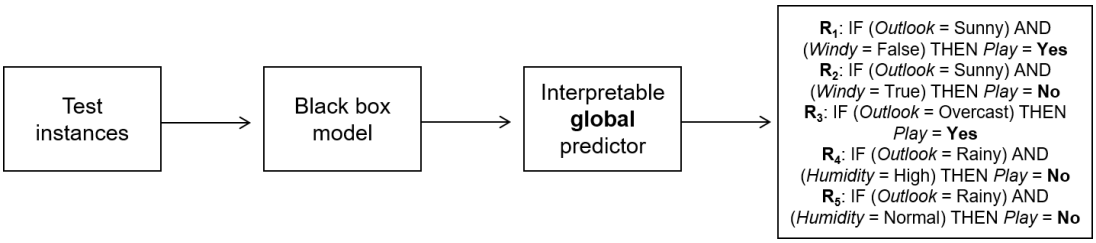


Figure 3.2: An example of the model explanation problem (from [41]).

While in the outcome explanation problem (Figure 3.1) we focus on explaining a single outcome that is obtained given a specific input (a data instance), in the model explanation problem (Figure 3.2), however, the need is to explain the overall behavior of the model (in this case, unlike in the first problem, a set of data instances is given). Finally, in the model inspection, the focus is to understand how the model behaves once the input changes.

A plethora of methods that tackle the black box explanation problem by providing explanations have been published [41]. These methods are often called as explanation methods (or explainers for short).

Explanation desiderata. A set of desirable properties for the explanation methods is given next. These properties can be further used to evaluate explanations of black box models [4] and also to select a explanation method [42, 48].

- *Fidelity* (or *faithfulness*) quantifies how well an explanation correctly captures the behavior of a black box model.
- *Stability* (or *robustness*) measures how well an explanation method does not change the explanations when the input changes, i.e. similar data instances must get similar explanations.
- *Interpretability* (or *comprehensibility*) indicates how readily it is for a human to understand explanations.

Due to the quantity and diversity of ML models and explainers available, we can arrange explanation methods in several ways. For instance, it is possible to categorize explanation methods based on: (1) the *data type* the methods support, (2) the type of *explanation output*, (3) the *explanation scope*, and (4) the origin of explanations.

1. **Data type.** Explanation methods can be designed to be data-specific (e.g. explanations for image [1], text [19], and tabular data [73]) or data-agnostic (e.g. LIME [72] and SHAP [59]).
2. **Explanation output.** The type of the explanation is another criteria to arrange the explanation methods. For instance, rule-based (e.g. Anchors [25, 73] and LORE [40]), example-based (e.g. prototype [51, 57] and counterfactuals [39, 85]), feature importance-based (e.g. LIME and SHAP), and concept-based (e.g. [36]) explanation methods.

3. **Explanation scope.** Explanation methods can also be divided based on the scope, i.e., what is being explained. In this case, the methods can be arranged into two main groups: (1) those that provide *local explanations* and (2) those that compute *global explanations*. Local explanation methods generate explanations for individual predictions, while global explanations give an understanding of the global behaviour of the model. This organization of explanations methods is linked to outcome explanation and model explanation problems (Figures 3.1 and 3.2). That is, local explanations can be used to tackle the outcome explanation problem, while global explanations must be used to cope with the model explanation problem.

4. **Explanation's source.** Methods to enhance explanability can also be arranged based on the origin of the explanations. That is, explanations can be obtained (1) directly from ML models once we are restricted to interpretable models [31] or (2) from explanation methods that produce explanations for predictions of a given (not necessarily interpretable) model, i.e. post-hoc explanations such as LIME and SHAP.

In this work, we are interested in model agnostic explanation methods, i.e, methods that can explain any prediction model (classifier). We thus focus on post-hoc explanations in particular LIME and SHAP that are presented next.

3.2 Local post-hoc explanations

Let f be the prediction model[17], x be a target data instance, and $f(x)$ be the prediction we want to explain. In order to explain $f(x)$, LIME and SHAP need to have access to a neighbourhood of x. Then they generate data instances around x by applying perturbations. While generating neighbors of x, these methods try to ensure interpretability by using interpretable versions of x (denoted as x') and of its neighbourhood.

A mapping function $h_x(x')$ is responsible for converting x' from the interpretable space to the feature space. For instance, different data types require distinct mapping functions h_x. For tabular data, h_x treats discretized versions of numerical features, while for textual data, it deals with the presence/absence of words.

Explanations take the form of surrogate models that are linear models (transparent by design) for LIME and SHAP. That is, they approximate f by learning a linear function g,

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i, \tag{3.1}$$

where ϕ_i are the weights of the models which correspond to the importance of features.

In addition, the function g should optimize the following objective function:

$$\arg\min_{g \in \mathcal{G}} \{\mathcal{L}(f, g, \pi_x) + \Omega(g)\}, \tag{3.2}$$

Ω measures the complexity in order to ensure (regularizer) interpretability of the linear model produced by the explainer. $\mathcal{L}$ is the loss function defined by:

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in Z} [f(z) - g(z')]^2 \pi_x(z), \tag{3.3}$$

[17] Also named as classifier, predictor or simply model.

where z is the interpretable representation of x, and $\pi_x(z)$ defines the neighborhood of x that is considered to explain $f(x)$.

Since LIME and SHAP share some commonalities, we present next a list of generic steps that both explanation methods used to produce local explanations for single predictions of a given model.

Common steps to both explainers LIME and SHAP

1. **Generate instances**. The first step is to generate instances by perturbation. That is, generated instances constitute a new set that will be further used to explain the prediction of the target instance x.
2. **Get predictions.** For each generated instance, the explainer (LIME or SHAP) uses the original prediction function f to obtain a prediction (e.g. class label).
3. **Build a linear model.** Then, the explainer fits a weighted linear model g on the obtained predictions by using Equation 3.2. Finally, the weights of g, i.e. ϕ, are returned as feature importance.

In spite of sharing the aforementioned commonalities, LIME and SHAP differ in the definition of the kernel π_x and also in the complexity function Ω used to produce explanations. In the following we recall each method and highlight their differences.

3.2.1 Local Interpretable Model Agnostic Explanations (LIME) [72]

LIME[18] is an explanation method that provides local explanations. It was designed to have the following desirable properties.

- *Model-agnostic*: the explanation method should be able to explain any model regardless its inner workings.
- *Model-interpretability*: explanations should be easy to understand and provide an understanding between features and outcomes.
- *Local fidelity*: the method must be faithful at least locally, i.e. it must capture the behavior of the model being explained at least within a specific neighborhood.

Explanations obtained from LIME take the form of surrogate linear models. LIME learns a linear model by approximating the prediction and feature values in order to mimic the behavior of ML model. To do so, LIME uses the following RBF (Radial Basis Function) kernel π_x to define the neighborhood of x and considered to explain $f(x)$:

$$\pi_x(z) = exp\left(-\frac{\delta(x,z)^2}{\sigma^2}\right),$$

where δ is a distance function between x and z, and σ is the kernel-width. LIME tries to minimize $\Omega(g)$ for reducing the number of non-zero coefficients in the linear model.

The description of the general version of LIME is given in Algorithm 1. It starts by initializing the neighborhood set Z of the target instance x (line 1). Then, LIME generates instances and

[18] Available at https://github.com/marcotcr/lime.

populates Z (lines 2-5). It first samples around x' (line 3). That is, it uses the interpretable version x' of x for sampling. LIME then perturbs x' to obtain a set of generated instances z'_i, $i \in \{1, ..., N\}$, that belong to an interpretable space [34]. Next, the explanation method asks for a prediction of z_i, which is the version of z'_i in the original space, by using the original prediction function f (i.e., the original model). In addition, the distance to x is computed to each instance z, which is done thanks to the kernel π_x. These generated instances z'_i along with their predictions $f(z_i)$ and distances $\pi_x(z_i)$ constitute the set Z (line 4) Finally, the algorithm fit a linear model g using the set Z (line 6) and it returns the weights ϕ of g as feature importances.

Algorithm 1: LIME [72]

Data: f: model; N: number of samples; x: data instance; x': interpretable version of x; π: similarity kernel; K: length of explanation.
Result: ϕ: weights.

1 $Z \leftarrow \{\}$
2 **for** $i \in \{1, ..., N\}$ **do**
3 $\quad z'_i \leftarrow \text{SampleAround}(x')$
4 $\quad Z \leftarrow Z \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$
5 **end**
6 $\phi \leftarrow \text{FitModel}(Z, K)$
7 **return** ϕ

Example. To illustrate, let us consider the example of the Adult dataset where the goal is to predict if a person earns ≥50k dollars a year. Figure 3.3 presents a LIME explanation to the classification of an instance from the Adult dataset. For instance, the value "Capital Gain"≤ 0.0 contributes 0.29 to the class $\leq 50K$, whereas the value "Relationship"= *Husband* contributes 0.15 to the class $> 50K$.

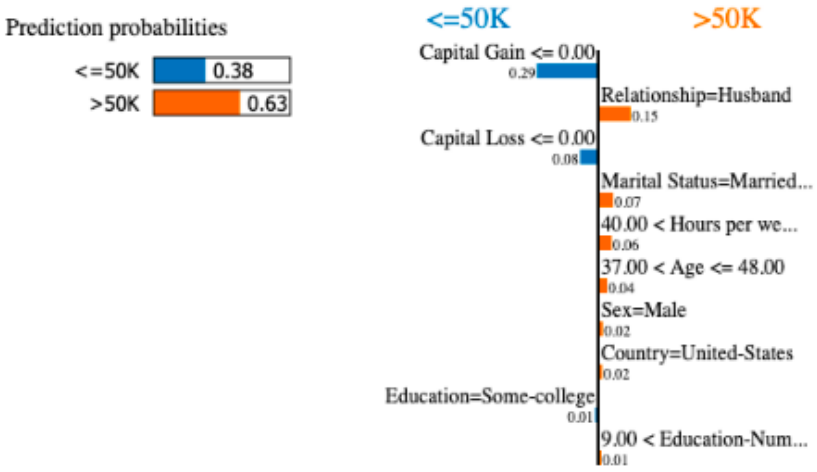


Figure 3.3: LIME explanation in case of Adult dataset

3.2.2 SHapley Additive exPlanations (SHAP) [59]

SHAP[19] is also an explanation method that provides local explanations, but that is based on coalitional game theory. It uses the notion of coalition of features to build a linear surrogate model g w.r.t. a single prediction $f(x)$ and whose coefficients ϕ correspond to the contributions of the features of x' (the interpretable (simplified) version of x). In the case of SHAP, these coefficients coincide with Shapley values [78]. Also, coalition of features defines the representation space for SHAP and it indicates which features are presented in x (see Figure 3.4).

Properties. SHAP was designed to attain the following properties.

1. *Local accuracy.* It states that the liner model g has to obtain the same output of f when g approximates f, i.e. for a given x and its interpretable version x',

$$f(x) = g(x').$$

2. *Missingness.* It states that features with missed values have no importance to the outcome.

$$x'_i = 0 \Rightarrow \phi_i = 0.$$

3. *Consistency.* Let us denote $f_x(z') = f(h_x(z'))$ and $z' \setminus i$ as $z'_i = 0$. Now, given two prediction functions f and f', consistency states that, if

$$f'_x(z') - f'_x(z' \setminus i) \geq f_x(z') - f_x(z' \setminus i)$$

then, $\phi_i(f', x) \geq \phi_i(f, x)$, for all inputs in the representational space (coalition space) $z' \in \{0, 1\}^M$.

Lundberg et al. [59] proposed a model-agnostic version of SHAP called KernelSHAP. Like LIME, KernelSHAP also minimizes the loss function described in Eq. 3.3, but it uses a different kernel and, unlike LIME, KernelSHAP sets $\Omega(g) = 0$ for the complexity of explanations.

Let $|z|$ be the number of present features in the coalition z, $\mathcal{M}$ be the maximum coalition size, and m be the number of coalitions. In order to attain the aforementioned properties of local accuracy, missingness, and consistency, the kernel function $\pi_x(z)$ used by KernelSHAP is defined as

$$\pi_x(z) = \frac{\mathcal{M} - 1}{\binom{\mathcal{M}}{|z|} |z| (\mathcal{M} - |z|)}.$$

Example. To illustrate, let us consider the example of the Adult dataset where the goal is to predict if a person earns ≥50k dollars a year. Figure 3.5 presents a SHAP explanation for a prediction using Logistic Regression classifier, where the Shapley value for "Capital Gain = 2,174" is around -0.15 indicating that this feature contributes to move the prediction towards the negative class.

[19] Available at https://github.com/slundberg/shap.

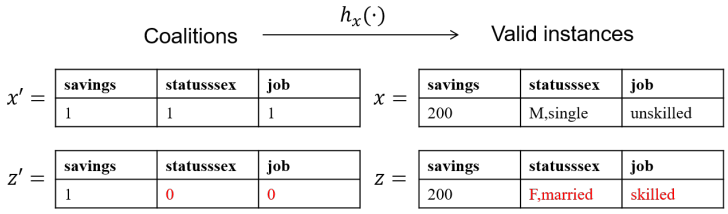


Figure 3.4: An example of a mapping function h_x that converts coalitions to valid instances (i.e., instances in the feature space).

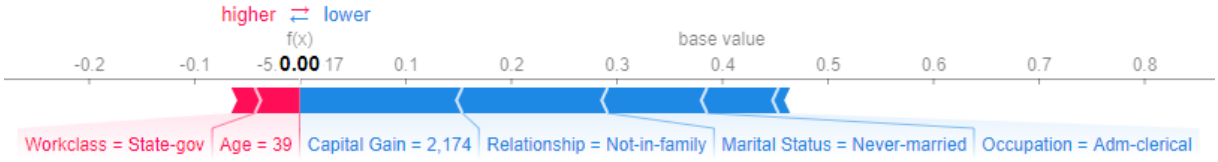


Figure 3.5: SHAP explanation of the prediction of an instance in the Adult dataset.

3.3 From local to global explanations

Both LIME and SHAP only provide explanations for individual predictions. However, in order to detect unintended biases, a global understanding of the inner workings of the ML model is may needed. Global explanation methods can be divided into two main groups: (1) methods based on a collection of local explanations [72], and (2) methods that provide representation based explanations [52]. In this section, we focus on the first group by recalling a strategy to obtain global explanations from individual explanations.

This strategy, so-called Submodular Pick sampling (SP), was proposed by Ribeiro et al. [72]. It consists of selecting a subset of local explanations that helps in interpreting the global behavior of a given ML model. SP was originally proposed to work along with LIME explanations and it is called SP-LIME. The main idea on which relies SP-LIME is to sample a set of instances whose explanations are not redundant and that has a "high covering" in the following sense.

Let B the desired number of explanations used to explain f globally, V a set of selected instances, I an array of feature importance, and W an explanation matrix –columns represent features and rows represent instances– that contains the importance (contribution) of features to each instance. SP-LIME picks instances that are explaining thanks to:

$$Pick(W, I) = \arg \max_{V, |V| \leq B} c(V, W, I), \tag{3.4}$$

where the function c defines the notion of non-redundant coverage which is calculated as

$$c(V, W, I) = \sum_{j=1}^{d'} 1_{[i \in V : \mathcal{W}_{ij} > 0]} I_j.$$

Example [72]. To illustrate how Algorithm 2 works, let us consider an example of a set of five data instances described by five features. For each instance, we ask for one explanation by using LIME as an explanation method (first loop, lines 1-3) and we get a vector with five values, i.e., the i-th value correspond to the importance of the i-th feature in the dataset. As a result, we get an explanation matrix W of dimensions 5×5. For simplicity, in this example, W is binary and it is presented bellow. Each line $(x_1, ..., x_5)$ corresponds to one data instance

Algorithm 2: Sub-modular pick algorithm [72]

Data: D: data instances; B: maximum number of explanations.
Result: V: selected explanations.

```
1  for x_i ∈ D do
2  |  W_i ← explain(x_i, x'_i)
3  end
4  for j ∈ {1, ..., d'} do
5  |  I_j ← sqrt(Σ_{i=1}^{n} |W_ij|)
6  end
7  V ← {}
8  while |V| < B do
9  |  V ← V ∪ arg max_i c(V ∪ {i}, W, I)
10 end
11 return V
```

and each column $(A_1, ..., A_5)$ represents one feature. Then, we compute the vector of features importance I (second loop, lines 4-6). For instance, the second feature A_2 is the one that has the highest importance since it explains most of data instances. Finally, the greedy choice is employed in the third loop (lines 8-10) and the algorithm stops when the budget B is reached. In this example, instances x_2 and x_5 are chosen (the two highlighted rows in the matrix bellow) since they cover almost all features, except the first feature A_1.

$$W = \begin{array}{c} \begin{matrix} A_1 & A_2 & A_3 & A_4 & A_5 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} \begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}$$

3.4 Drawbacks and pitfalls of employing LIME and SHAP

Explanations can give insights about the inner workings of opaque classifiers. However, it has been shown that LIME and SHAP present several drawbacks. They lack of stability. That is, slight perturbations in a data instance can completely change the explanation of the prediction. These methods are not computationally inefficient. That is, we need to do a hyper-parameter tuning in order to get suitable explanations. Also, explanations are also inconsistent. Running the same explanation method with the same parameters multiple times leads to different explanations which presents important randomness that might degenerate explanations.

Explanations based on feature importance can be manipulated. These manipulations can be done in such a way that biases linked to sensitive features are concealed while explanations show that classifier's outcomes do not depend on sensitive features. This problem has been pointed out in recent papers [26, 81, 82]. Explanations can be manipulated through adversarial attacks by exploiting the lack of stability and fidelity of the explanation methods in the following way. An adversarial model is applied to train a biased classifier based on a particular fairness notion. An auditor uses explanations based on feature importance to assess unfairness. LIME and SHAP generate data instances in the neighborhood of a data instance of interest in order to

obtain local explanations. Generating data instances is done by perturbation and the distribution of generated data is not the same as the distribution of input data. Once adversarial models can differentiate between the two distributions, they can take advantage of it in the way they train/modify a classifier to be biased, in order to fool LIME and SHAP explanations.

3.5 LIME and SHAP extensions

In this section, we present some extensions of LIME and SHAP that try to overcome the issues mentioned in the previous section. For instance, some extensions focused on improving fidelity, e.g. QLIME [13] and ILIME [79], while other extensions tackle the problem of instability, e.g. DLIME [88], and other extensions focus on locality and fidelity to improve LIME, e.g. s-LIME [35].

Here, we focus on extensions that either consider uncertainty when producing explanations or that allow ML practitioners to easily modify an explanation method to their needs. In this regard, we briefly present the following extensions.

- **build LIME yourself (bLIMEy)** [83]

  This extension tackles technical problems related to the implementation of LIME. That is, if ML practitioners decide to overcome some of LIME's issues, they need to fully understand the original source code of LIME so that they make direct modifications to the code, which may be a hindrance for some.

  bLIMEy allows ML practitioners to adapt different surrogate models for obtaining explanations (including LIME). As it is not possible to have an explanation method that outperforms all others in all scenarios, the idea is to provide a modular framework that allows ML practitioners to build their own surrogate models in order to obtain explanations adapted for their tasks. bLIMEy is essentially composed by three modules as follows.

  1. *Data representation.* The first module is responsible for converting the data from the original space to an interpretable space (and to do the opposite as well). That is, creating an interpretable version of the data instances whose predictions we want to explain. This process is required for image and textual data, while it is optional for tabular data.
  2. *Data sampling.* Generated data instances are needed for fitting a model afterwards. In this module, bLIMEy allows practitioners to choose an approach that samples instances. For instance, with bLIMEy we do not need to use the interpretable space for sampling instance in the case of tabular data, while
  3. *Explanation generation.* Here, unlike LIME, bLIMEy allows practitioners to choose a linear model to generate explanations. For instance, it can be either a linear regression or a decision tree, while in the original LIME code it is fixed.

- **BayesLIME and BayesSHAP** [80]

  BayesLIME and BayesSHAP are instantiations of a Bayesian framework that is designed to provide local explanations with the associated uncertainty. Slack et al. [80] argue that taking into account uncertainty when generating explanations help to cope with the issues mentioned previously, i.e. instability, inconsistency, randomness, and computational inefficiency. They thus introduced the following techniques.

1. A Bayesian method that generates local explanations that also captures the associated uncertainty, i.e. the method outputs feature importance as a point along with its correspondent interval. Instead of estimating feature importance as weights of a linear model, they considered feature importance as a distribution.
2. An approach to estimates the number of perturbations. That is, they estimate the number of perturbations to generate explanations with a minimal confidence.
3. A sampling procedure that reduces the computational cost of perturbations. This procedure is inspired by active learning techniques that try to maximize the gain of information when selecting data instances. Here, they focus on on perturbations with high uncertainty, instead of doing a random sampling.

They showed that the instantiations of this Bayesian framework, i.e. BayesLIME and BayesSHAP, were capable of estimating feature importances with high confidence that actually fall within a given interval. Although Slack et al. [80] argue that they tackle several issues at the same time with the Bayesian framework, their approach is still vulnerable to attacks if the same method of LIME and SHAP for sampling data instances is employed.

3.6 PathExplain [47]

In this section, we present an explanation method that, unlike LIME and SHAP, is model-specific. This explainer is called PathExplain and it is focused on explaining differentiable models, such as deep learning models. More precisely, Path Explain[20] estimates feature importance thanks to Integrated Gradients, which is based on the Aumann-Shapley value, a variant of the Shapley value for continuous players. Given a model f and a data instance x to be explained, the contribution Φ_i of the i-th feature w.r.t. x is computed as

$$\Phi_i(x) = (x_i - x_i^*) \int_{\alpha=0}^{1} \frac{\partial f(x^* + \alpha(x - x^*))}{\partial x_i} d\alpha,$$

where x' is a baseline instance, i.e., a vector of base values. Base values are used by explanation methods to take into consideration the lack of information when explanations are being generated. The idea is to vary the values from x^* to the instance we want to explain x. Hence, f must be a differentiable function between x^* and x (not necessarily a deep learning model). This process may change the output since we are modifying the input. As a result, the explanation (feature contribution) for the i-th feature of $f(x_i)$ is the mean variation of the curve that is being integrated weighted by the interval between x_i and x_i^*.

[20] Available at https://github.com/suinleelab/path_explain.

Part II

Contributions

Chapter 4

Mitigating unfairness through unawareness

Contents

4.1 Process fairness meets explanations 44

4.2 Components of FixOut 44

4.2.1 Fairness assessment 45

4.2.2 Unfairness mitigation: the Ensemble Building component 46

4.3 FixOut the algorithm 49

4.4 Example 51

4.5 Discussion 51

In this chapter, we introduce FixOut (FaIrness through eXplanations and feature dropOut), a framework that tackles the two main problems of algorithmic fairness, i.e., (1) fairness assessment and (2) unfairness mitigation. FixOut uses explanations to assess process fairness in the fairness assessment phase. In order to mitigate unfairness, the framework pushes further fairness through unawareness by combining two different fairness interventions without compromising classification performance. FixOut removes sensitive features before training classifiers and modifies the input which characterizes the pre-processing phase. The framework produces a pool of classifiers whose outputs are combined thanks to an aggregation function. This function manipulates classifiers' decision borders in order to enforce fairness, which characterizes the post-processing phase.

This chapter is organized as follows. We start by connecting process fairness and explainability in Section 4.1, i.e., we then present an overview of the components of FixOut in Section 4.2. In Section 4.2.1, we detail how FixOut assess fairness by employing local post-hoc explanations in the first component of the framework. We then present FixOut as a fairness processor in Section 4.2.2, i.e., the second component of the framework which is responsible for mitigating unfairness by combining pre-processing and post-processing fairness interventions. We conclude this chapter by presenting a full example of applying FixOut in a real world dataset in Section 4.4 and discussing the advantages, limitations and perspectives for the generic framework in Section 4.5.

4.1 Process fairness meets explanations

Explanations can help to reveal biases of algorithmic decisions. In [12] it was delivered a potential solution to deal with process fairness in ML classifiers. The key idea was to use the explanation method LIME to assess whether a given classifier was fair by measuring its reliance on salient or sensitive features. This component was then integrated in a human-centered workflow called LimeOut, that receives as input a triple $(M, D, \mathcal{A}^*)$ of a classifier M, a dataset D and a set $\mathcal{A}^*$ of sensitive features, and outputs a classifier M_{final} less dependent on sensitive features without compromising accuracy. To achieve both goals, LimeOut relies on *feature dropout* to produce a pool of classifiers that are then combined through an ensemble approach. Feature dropout receives a classifier and a feature A as input, and produces a classifier that does not take A into account [12].

Empirical studies [12] showed that LimeOut's ensemble models are less dependent on sensitive features when compared to original models. However, several issues concerning the use of explanation methods for assessing process fairness have been recently raised. For instance, [26,81] questioned the usefulness of explanations to assess fairness by showing that it is possible to perform "adversarial attacks" to modify explanations in order to conceal unfairness issues. This led to a thorough empirical investigation [6] beyond process fairness, and where LimeOut showed consistent improvements with respect to widely used fairness metrics such as disparate impact, equal opportunity, demographic parity, equal accuracy, and predictive equality. In [6] it was also claimed the adaptability of LimeOut to other data types as well as to other explanation methods. This is particularly relevant given the drawbacks of LIME explanations that have been pointed out in the literature [34,65].

Here, we tackle the latter issues by showing that LimeOut can be adapted to different explanation methods and aggregation functions, and by empirically observing beneficial impacts on widely used fairness metrics. More precisely, we propose FixOut, a framework that extends LimeOut by allowing any explanation method based on feature importance. To illustrate, we consider FixOut instantiated by SHAP and LIME[21] to assess model fairness. Also, to construct the final ensemble model, FixOut can employ either a simple average as aggregation rule or a weighted average to take into account the global contributions of sensitive features.

4.2 Components of FixOut

FixOut has two main components, each one associated with one problem of algorithmic fairness. The first one is responsible for assessing fairness of a pre-trained classifier, while the second one is in charge of mitigating unfairness. Figure 4.1 illustrates these two components and their interactions. First, FixOut starts by assessing fairness of a pre-trained model in the first component. Then, if this model is deemed unfair, the framework triggers the second component, whose goal is to produce another model of the same type as the pre-trained one but less reliant on sensitive features.

More precisely, the framework FixOut receives a triple $(M, D, \mathcal{A}^*, E)$ of a pre-trained classifier M, a dataset D, a set of sensitive features $\mathcal{A}^*$, and an explanation method E based on feature importance. Again, FixOut starts by applying the first component "Fairness assessment" using E as the explanation method. For instance, it can employ either SHAP, LIME or any other method that estimates feature importance, and thus to evaluate the dependence of M

[21] The instantiation of FixOut with LIME explanations and the simple average aggregation is equivalent to LimeOut.

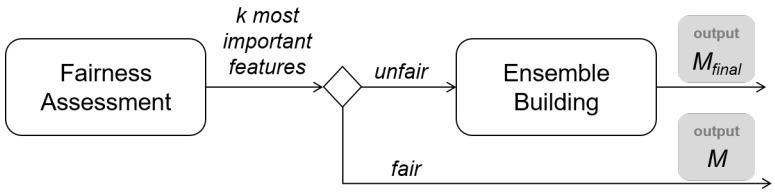


Figure 4.1: The main components of FixOut. (1) In the fairness assessment phase, the goal is to list the top-k most important features and if there exists sensitive features among the top-k the model is deemed unfair. (2) In the ensemble building component, FixOut works as a fairness processor by combining two fairness interventions: pre-processing and post-processing.

on sensitive features (see Figure 4.5). The output of "Fairness assessment" is a list $L^{(k)}$ of the k most important features $A_1, A_2, \ldots, A_k$. The framework applies the following rule to decide whether M is fair:

if $L^{(k)}$ contains sensitive features $A_{j_1}, A_{j_2}, \ldots, A_{j_i}$ in L with $i \geq 1$, then M is deemed unfair and the FixOut's second component applies; otherwise, it is considered fair and no action is taken.

4.2.1 Fairness assessment

The first problem that FixOut tackles is to assess fairness of a pre-trained model M. To do so, the framework relies on explanations as LimeOut does. Yet, unlike LimeOut, FixOut is not explainer-specific, i.e., it supports any explanation method that provides explanations in the form of feature importance.

In order to unveil unfairness of a ML model, FixOut needs explanations that globally explain the model. However, explanation methods such as LIME and SHAP provide explanations for individual predictions, i.e., local explanations. In order to overcome this issue, FixOut either uses one of the two following approaches to obtain global explanations from a set of local explanations.

- **Random Sampling (RS).** This is a straightforward approach that randomly select instances based on a uniform distribution. Note that, in the presence of highly imbalanced datasets, this approach can select more instances (and by extent their explanations) of a particular class than another class.
- **Submodular Pick sampling (SP).** Here, data instances are selected by following the sampling procedure detailed in Section 3.3, which focuses on data instances whose explanations are representative and are not redundant. Note that, sensitive and non-sensitive features are taken into account for this sampling strategy.

Once the sampling procedure is finished, all local explanations must be aggregated to obtain one single explanation for the whole model, i.e. a global explanation. As we have multiple values that represent the importance of one particular feature for predictions of different data instances, More precisely, for each feature A_j, FixOut sums the contribution of A_j from all local explanations this feature appears. At the end, the framework obtains a list of feature importance for all features that appear in the explanations of the selected set of data instances.

An example of the output from the fairness assessment phase is presented in Table 4.1. Here, SHAP explanations are employed. On the left side, we have the top-10 most important features

(in a descending order w.r.t. the absolute value of the contribution) of the pre-trained model. Note that two sensitive features (highlighted lines) are among that list. Also, the most important feature, i.e. the one the model is the most dependent for, is the sensitive feature "statussex". On the right side, the top-10 most important features is obtained from the ensemble model built by FixOut. It is noteworthy that the contributions of sensitive features are smaller in the right side. This indicates that the model provided by FixOut is (globally) less dependent on those sensitive features.

Table 4.1: An example of global explanations for Random Forest on the German dataset.

Original (SHAP)

Feature	Contrib.
statussex	-10.758909
property	10.676458
credithistory	10.264842
residencesince	8.108638
employmentsince	6.818476
existingchecking	6.308758
housing	-5.649528
installmentrate	5.125154
duration	4.838629
telephone	3.981387

FixOut using SHAP

Feature	Contrib.
credithistory	13.246923
employmentsince	9.550589
property	8.958718
residencesince	7.264848
installmentrate	5.893469
housing	-3.829469
statussex	-3.313667
duration	2.678797
existingcredits	2.412488
telephone	2.269493

4.2.2 Unfairness mitigation: the Ensemble Building component

The second problem that FixOut tackles is to reduce unfairness of a pre-trained model M in the case M is deemed unfair. To do that, the framework applies a hybrid-processing fairness intervention. That is, it uses more than one fairness process to enforce fairness. In the case of FixOut, the framework takes advantage of two different types of interventions: pre-processing and post-processing. The choice of using this two types of fairness-enhancing interventions is linked to our desire of proposing an approach that is model-agnostic. In connection with this goal, pre-processing and post-processing are suitable options since they are usually model-agnostic, while employing in-processing fairness intervention would force us to know the inner workings of ML models.

Figure 4.2 illustrates the proposed framework as a hybrid fairness-enhancing approach. In the following sections, we detail each fairness intervention applied by FixOut in order to build a new model M_{final} from M.

Pre-processing step: feature dropout

To make M fairer, the first step which FixOut follows is to apply a pre-processing fairness intervention based on fairness through unawareness. That is, the framework eliminates sensitive features before training a model, i.e. feature dropout. Unlike the straightforward approach of fairness through unawareness, FixOut combines feature dropout with an ensemble approach. The main idea behind the combination of fairness through unawareness and an ensemble approach is to cope with the tension between fairness and classification performance.

More precisely, FixOut employs feature dropout in the following way. Given i sensitive features in the top-k most important features, the framework uses the i sensitive features $A_{j_1}, A_{j_2}, \ldots, A_{j_i} \in L^{(k)}$ to build a pool of $i+1$ classifiers in the following way:

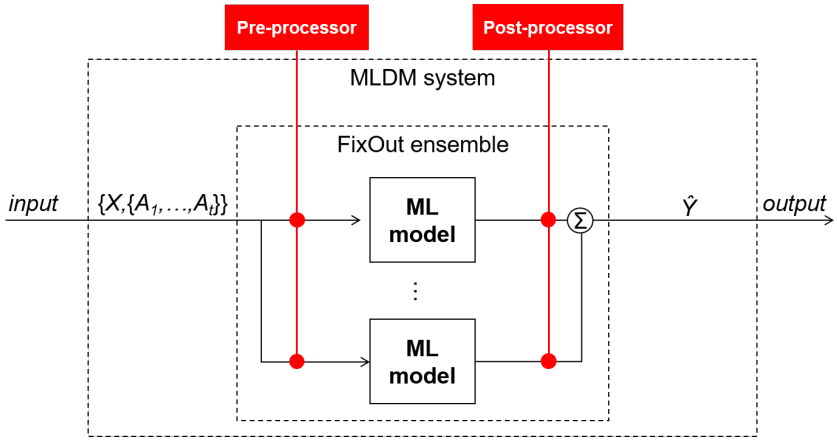


Figure 4.2: FixOut as a hybrid fairness processor. Two interventions are applied: (1) in the pre-processing step, the input is modified based on the approach fairness through unawareness (sensitive features are removed before training and testing) and (2) in the post-processing step, an aggregation rules is used to manipulate the decision border

- for each $1 \leq t \leq i$, FixOut trains a classifier M_t after removing A_{j_t} from D,
- and an additional classifier M_{i+1} trained after removing all sensitive features in $L^{(k)}$ from D.

This pool of classifiers is then used to construct an ensemble classifier M_{final}.

Note that if only one sensitive feature appears in $L^{(k)}$, i.e. $i = 1$, FixOut will produce an ensemble model with only one classifier which is trained without the sensitive feature. On the one hand, the single classifier is fairer than the pre-trained model w.r.t. process fairness. On the other hand, the classification performance of the produced ensemble may be compromised, since the single classifier is trained with less information.

Figure 4.3 depicts an example of an ensemble model built by FixOut in the pre-processing step. In this example, three sensitive features are taken into account ($i = 3$), which generates a pool with four classifiers. The first classifier is trained after removing "MaritalStatus". The second classifier is trained once "Sex" is removed. The third classifier is trained by ignoring "Race". Finally, the last classifier is trained without any of the three sensitive features.

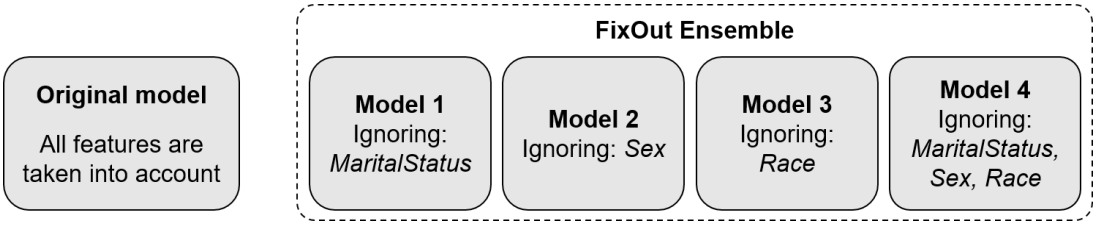


Figure 4.3: Illustration of which sensitive features are taken into account by each model.

Pitfalls of fairness through unawareness. Although removing sensitive features is an straightforward pre-processing approach that helps to improve fairness, it may lead to side-effects. Here, we point out the following problems when using fairness through unawareness.

- *Classification performance.* One of the first problems that we may face when only fairness through unawareness is that classification performance may drop [12]. This can be

explained since classifiers have less information.

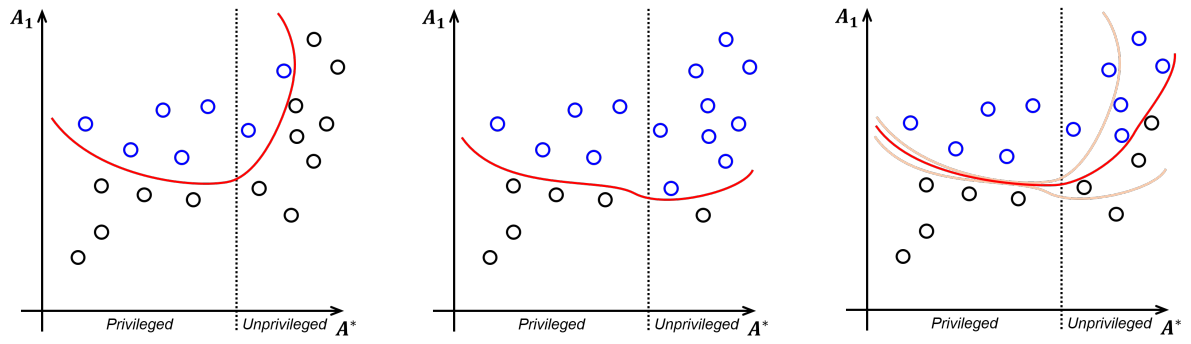
- *Proxy features.* Another problem is the presence of proxy features. These non-sensitive features are highly correlated with sensitive features. Even though sensitive features are not present in the dataset after removing them, they can help classifiers to indirectly discriminate. Note that, in the case of FixOut with the presence of proxy features, ML practitioners have to indicate for each sensitive feature $A^*_{j_t}$ which features work as proxies to $A^*_{j_t}$.

- *Dimensionality.* In high-dimensional spaces –in the presence of a high number of features– feature dropout may face another problem: the effectiveness of this approach decreases. Dropping out one sensitive feature among a large list of features has less impact when there is a small list of features. This is potentially problematic in some particular problems, e.g., on textual data [5].

Post-processing step: aggregation

As a post-processing fairness processor, FixOut supports any function that receives probability scores and outputs a binary class label. Here, we thus propose three different aggregation functions, namely: *simple*, *weighted* and *learned weighted* averages. The idea is that these functions manipulate classifier's outputs in order to enforce fairness. Note that FixOut uses only classifiers that provide probabilities.

Figure 4.4 depicts an example of how the aggregation function changes the decision boundary and impacts an unprivileged group. M_1 and M_2 are models belonging to a pool of classifiers. Each model is trained on a different subset of features. Precisely, M_1 does not take the sensitive feature A^* into account, while M_2 does consider A^* to produce its outcomes. M_{final} is an ensemble model that essentially aggregates the outputs of M_1 and M_2. Note that, M_{final}'s decision boundary is a compromise between M_1 and M_2's decisions boundaries. Also, M_{final}'s decision boundary is fairer than M_1 since it is not use A^* for predicting.



(a) M_1 (b) M_2 (c) M_{final}

Figure 4.4: An example of the impact of an aggregation function on an unprivileged group. M_1 and M_2 are single models while M_{final} is an ensemble that combines M_1 and M_2 outputs thanks to an aggregation function. Blue dots indicate positive class while black dots indicate negative class. The decision boundary is indicated by the red line while the definition of sub-populations is made by the dashed line, which is based on the sensitive feature A^*.

Simple average. This is an immediate solution for aggregating classifiers' outputs. Here, all outputs have the same importance, even though some classifiers might be fairer than others. Given a data instance x and a class C, for an ensemble classifier M_{final} that uses simple averaging, the probability of x being in class C is computed as follows

$$\mathbb{P}_{M_{final}}(x \in C) = \frac{1}{i+1} \sum_{t=1}^{i+1} \mathbb{P}_{M_t}(x \in C), \tag{4.1}$$

where $\mathbb{P}_{M_t}(x \in C)$ is the probability predicted by model M_t.

Weighted average. This function assigns a different importance for classifiers' outputs. In order to to so, the contributions of sensitive features are taken into consideration. Let $c'_{j_t} \in [0,1]$ be the normalized global feature contribution associated with ${A^*}_{j_t}$. We standardize feature contributions by $c'_{j_t} = \frac{c_{j_t} - min(L^{(k)})}{max(L^{(k)}) - min(L^{(k)})}$, where $min(L^{(k)})$ and $max(L^{(k)})$ are the lowest and the highest feature contribution among $L^{(k)}$, respectively. Now, let us define the weights w_t of M_t and the weight w_{i+1} of M_{i+1} as

$$w_t = \frac{c'_{j_t}}{1 + \sum_{u=1}^{i} c'_{j_u}},\ 1 \leq t \leq i, \quad \text{and} \quad w_{i+1} = \frac{1}{1 + \sum_{u=1}^{i} c'_{j_u}}.$$

The main idea behind using feature contribution in the weighted average is to ensure higher weights for classifiers trained without sensitive features whose contributions to M's outcomes are high. Also, the additional classifier M_{i+1}, the one that is trained without any sensitive feature, receives a higher weight. For an ensemble classifier M_{final} that uses weighted averaging, the probability of x being in class C is computed as follows:

$$\mathbb{P}_{M_{final}}(x \in C) = \sum_{t=1}^{i+1} w_t \mathbb{P}_{M_t}(x \in C). \tag{4.2}$$

Learned weighted average. The third aggregation function also assigns different importance for classifiers' outputs. However, unlike the weighted average, weights are learned by a learning algorithm, e.g. Logistic Regression, instead of using directly contributions of sensitive features.

For each data instance, we ask the probabilities from each classifier in the pool. We then associate the list of probabilities obtained from all classifiers with the actual label. Thus, we have a dataset where each data instance is a list of probabilities with its label. This new dataset allows us to train a logistic regression classifier. After training, we use as weights the coefficients from the trained classifier. The following example illustrates how FixOut works.

4.3 FixOut the algorithm

Algorithm 3 summarizes the framework FixOut as a single algorithm. The function Global-Explanations (line 1) encodes the first component of the framework, which was detailed in Section 4.2.1. The function Train (lines 4 and 5) triggers the second component of FixOut. This function is detailed in Section 4.2.2 where we explained how FixOut combines two fairness interventions in order to reduce unintended biases w.r.t. two or more sensitive features.

Table 4.2: SHAP global explanations for Random Forest classifiers on German dataset

Original		Ensemble with *simple* average		Ensemble with *weighted* average	
Feature	**Contrib.**	**Feature**	**Contrib.**	**Feature**	**Contrib.**
statussex	-10.758909	credithistory	13.246923	credithistory	19.144550
property	10.676458	employmentsince	9.550589	property	10.470889
credithistory	10.264842	property	8.958718	employmentsince	7.097584
residencesince	8.108638	residencesince	7.264848	residencesince	6.916727
employmentsince	6.818476	installmentrate	5.893469	savings	-6.031979
existingchecking	6.308758	housing	-3.829469	installmentrate	4.795154
housing	-5.649528	**statussex**	-3.313667	housing	-3.207251
installmentrate	5.125154	duration	2.678797	existingcredits	2.883065
duration	4.838629	existingcredits	2.412488	duration	2.857096
telephone	3.981387	**telephone**	2.269493	purpose	2.123851

Algorithm 3: FixOut

Data: M: pre-trained model; D: dataset; $\mathcal{A}^*$: sensitive features; E: explainer; k: number of features considered.

Result: M_{final} an ensemble model if M is considered unfair.

$L \leftarrow \text{GlobalExplanations}(E, M, D, k)$
$\mathcal{A}' \leftarrow \{A_t \mid A_t \in \mathcal{A}^* \wedge A_t \in L\}$
if $|\mathcal{A}'| > 0$ **then**
    $M_{final} \leftarrow \{\text{Train}(M_t, D - \{A_t\}) \mid A_t \in \mathcal{A}'\}$
    $M_{final} \leftarrow M_{final} \cup \text{Train}(M_{t+1}, D - \mathcal{A}')$
    return M_{final}
else
    return M

4.4 Example

We illustrate FixOut on the Adult dataset. The goal is to predict if an American citizen earns more than 50k dollars per year based on census information. In this dataset, the sensitive features are "MaritalStatus", "Race", and "Sex".

The global explanations and pool of classifiers obtained from the experiment are depicted in Figure 4.5. In the right side of the figure, we can see the ranking of features' contributions $L^{(k)}$, where $k = 10$ (also referred to here as the top-10 most important features), for both pre-trained (original model) and FixOut's ensemble classifiers. In the lower left part of the same figure, the pool of classifiers is shown. Note that, as we considered three features as sensitive features, FixOut trains four classifiers: three classifiers are trained without one sensitive feature (either "MaritalStatus", "Race", or "Sex") and a fourth one without any sensitive feature (all three sensitive features are removed before training).

Global explanations of the pre-trained classifier show that this classifier is dependent on sensitive features, i.e., all sensitive features have (absolute value of) contribution that place them in the top-10 most important features. On the other hand, global explanations of FixOut's ensemble show that the pool of classifiers obtained from feature dropout is less reliant of sensitive features. Note that, only "MaritalStatus" appears in the top-10 and this sensitive feature has lower contribution for the ensemble's outcomes than for the pre-trained classifier's outcomes.

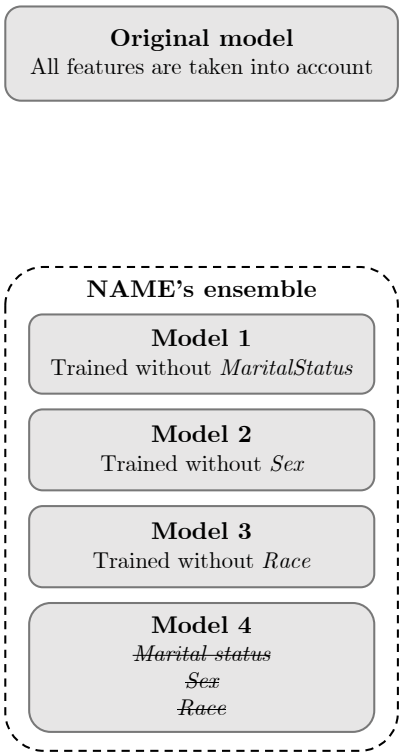
4.5 Discussion

In this chapter, we have proposed FixOut a human-centered and model-agnostic framework to make ML models fairer. FixOut is proposed to address and tackle process fairness: it first assesses the dependence of a given pre-trained ML model on sensitive features by global explanations in the form of feature contributions to the model's outcomes. If the model is shown to rely on sensitive features, then FixOut employs feature dropout followed by an ensemble approach to produce a new model.

Our proposed framework is a generic ML pipeline that is focused on binary classification. We instantiate this generic approach in the following chapters. Yet, in this section, we discuss straightforward limitations of our approach and natural perspectives of future work.

- *Utility.* Despite the fact that one can use FixOut in the presence of only one sensitive feature in the top-k, its utility is reduced in this particular scenario. That is, in the presence of only one sensitive feature, the obtained ensemble comprises only one classifier which is trained without the sensitive feature. We do not expect to find a suitable compromise between fairness and classification performance, in this case, by only removing a feature. For instance, classifiers trained without one (or more than one) sensitive feature(s) often present lower classification performance alone when compared with a pre-trained model, the one trained with all features. The main goal of employing an ensemble approach in FixOut is to tackle the fairness-accuracy trade-off in the presence of multiple sensitive features, as the aggregation of multiple classifiers helps to maintain the classification performance. In real-world scenarios, more than one sensitive feature is often found in tabular datasets (e.g. the experiments described in Chapter 5).

- *Adaptability.* The framework could be extended to classification problems that have more than two classes. Also, the idea of using ensemble models with feature dropout can also be



Feature (top 10)	**Contribution**
CapitalGain	8.133364
Education-Num	3.180944
Age	1.824722
MaritalStatus	1.354002
Hoursperweek	0.767099
Education	-0.518881
Country	-0.345699
Race	0.203788
Sex	0.178965
Occupation	0.064353

Feature (top 10)	**Contribution**
CapitalGain	7.877460
Education-Num	3.047071
Age	2.572467
Hoursperweek	1.193460
Country	-0.881120
RelationShip	0.223869
Workclass	-0.214024
fnlwgt	0.130579
MaritalStatus	0.120827
Education	0.081567

Figure 4.5: Impact of FixOut on global explanations of original model (pre-trained classifier) and FixOut's ensemble classifier. Example taken from an experiment on the Adult dataset using a bagging ensemble as pre-trained classifier and LIME.

applied to regression problems. Different forms of explanations, e.g. Anchor explanations, are not supported by FixOut.

- *Computational cost.* FixOut needs to re-train ML models to produce and output an ensemble model as a fairer model. Even though the number of models to be re-trained is linear w.r.t. the number of sensitive features, re-training models can be very costly in practice in the presence of huge and complex pre-trained models.

Chapter 5

Mitigating unfairness on tabular data: the first application case of FixOut

Contents

5.1 Tabular datasets 54
5.1.1 German 55
5.1.2 Adult 55
5.1.3 LSAC 55
5.2 Experimental setup 56
5.3 Experimental results 56
5.3.1 Process fairness assessment 56
5.3.2 Classification performance assessment 58
5.3.3 Fairness metrics assessment 59
5.4 Automating the choice of FixOut's parameter 62
5.4.1 Selection of instances to assess fairness 63
5.4.2 Experimental evaluation of Find-K 64
5.5 Discussion and perspectives 66

In this chapter, we instantiate FixOut for tackling the problem of unfairness mitigation on models trained on tabular data. Our main goal is to verify whether FixOut is able to reduce unfairness, from the lens of process fairness, without compromising classification performance. As FixOut is a generic framework that requires ML practitioners to make choices such as the explanation method, the maximum number of features to be considered important and the type of aggregation function, we define four instances of FixOut, namely: F+SHAP, Fw+SHAP, F+LIME, and Fw+LIME. They are briefly described bellow.

- F+SHAP uses SHAP as explanation method and simple average as aggregation function
- F+LIME uses simple average, similarly to F+SHAP, but with LIME explanations
- Fw+SHAP also employs SHAP but instead of applying the simple average rule, it employs the weighted average function described in Section 3.3
- Fw+LIME uses LIME as explainer but with the weighted average function.

These four instances of FixOut are used in the experiments in order to answer the following questions

Q1 Are there differences between FixOut with LIME explanations and FixOut with SHAP explanations?
We tackle the unfairness issues showing that LimeOut can be adapted to different explanation methods and aggregation functions. FixOut extends LimeOut by allowing any explanation method based on feature importance. To illustrate, we consider FixOut instantiated by SHAP and LIME to assess process fairness.

Q2 What is the impact of aggregation function in the performance of FixOut's output models?
To construct the final ensemble model, FixOut can employ different aggregation functions (e.g., simple average or weighted average). We instantiate FixOut with one of the aggregation functions presented in Section 4.2.2.

Q3 Does FixOut improve standard fairness metrics?
We designed FixOut to improve process fairness, which is a particular fairness notion. However, the literature reports various definitions of fairness, we thus evaluate FixOut on the well known standard fairness notions to verify empirically if we also observe beneficial impacts on those metrics.

Q4 How to reduce human intervention in FixOut usage on tabular data?
FixOut uses model-agnostic explanation methods to assess process fairness. It requires beforehand that users define the sampling size of data instances and the number of features used to assess fairness of pre-trained models, which raises the need for automation or, at least, fine-tuning. We then propose an algorithm that automatically selects the number of features used by FixOut to assess fairness.

This chapter is organized as follows. We start by describing the tabular datasets we used during the experiments in Section 5.1. We then explain the pre-processing steps and the experimental setup we applied in Section 5.2. Next, in Section 5.3, we analyze the experimental results from different perspectives, namely: process fairness assessment, classification performance and fairness metrics assessment. In Section 5.4, we propose a method for automating the choice of the parameter k required by FixOut and we also present the experimental results obtained when the proposed method is used with FixOut. Finally, we end this chapter by discussing future work for these instantiations of FixOut (Section 5.5).

5.1 Tabular datasets

The experiments on tabular data were performed on 3 real-world datasets, which contain sensitive features and are briefly outlined below. The choice of datasets and sensitive features were mostly done based on the literature on fairness. That is, we selected datasets (and sensitive features) that have also been used by the fairness community. For each dataset, non-sensitive features that are numerical or binary are used directly in the classification process without any pre-processing operation. In order to identify cases of highly correlated sensitive and non-sensitive features and thus avoid the *red-lining effect* [18] before performing any experiment, a correlation analysis was performed and is available in Appendix A. We did not find any pair (sensitive, non-sensitive) of features that is highly correlated in any of the datasets used in the experiments described

here (see Appendix A). Note that in the presence of highly correlated pair of features, non-sensitive features must be input along with sensitive features so that FixOut takes them into consideration during the pre-processing step, i.e. when the framework applies feature dropout (Section 4.2.2).

Table 5.1: Tabular datasets used in the experiments.

Dataset	# features	# instances	Sensitive features
German	20	1000	"statussex", "telephone", "foreign worker"
Adult	14	32561	"MaritalStatus", "Race", "Sex"
LSAC	11	26551	"race", "sex", "family_income"

5.1.1 German

This dataset contains 1000 instances and it is available in the UCI repository[22]. For this dataset, the goal is to predict if the credit risk of a person is good or bad. Each applicant is described by 20 features in total out of which 3 features are considered as sensitive.

We consider "statussex" as sensitive feature, allowing us to divide the population into male applicants and female applicants. Similarly, "foreign worker" is treated as sensitive; in this case, the population is divided into foreign applicants and non-foreign applicants. Finally, we consider "telephone" as sensitive feature. Like the previous sensitive features, the applicants are divided based on whether they provided their phone number or not (unprivileged group). The reason why we consider "telephone" as sensitive is that companies may use this information for other purposes (e.g. marketing campaigns) than credit risk evaluation; in addition, the presence of absence of the phone number should not change the credit risk of a person.

5.1.2 Adult

This dataset is also available in the UCI repository[23] that is also known as "Census Income" dataset. The task is to predict whether the salary of an American citizen exceeds 50 thousand dollars per year based on census data. It contains more than 32000 instances. Each data instance is described by 14 features out of which 3 features are considered as sensitive.

We consider first "sex" a sensitive feature, allowing us to divide the population between men and women . Then, "MaritalStatus" is also treated as sensitive feature; in this case, the population is divided into two groups based on whether they are married with a civilian spouse (`Married-civ-spouse`) or not in the dataset (`Divorced`, `Never-married`, `Separated`, `Widowed`, `Married-spouse-absent`, and `Married-AF-spouse`). Finally, we also treat "race" as sensitive where white people are compared to non-white people (`Black`, `Asian-Pac-Islander`, `Amer-Indian-Eskimo`, and `Other`).

5.1.3 LSAC

This dataset contains information about 26551 law students[24]. The task is to classify whether law students pass the bar exam based on information from the study of the Law School Admission Council (LSAC) that was collected between 1991 and 1997. Each student is characterized by

[22] https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)
[23] http://archive.ics.uci.edu/ml/datasets/Adult
[24] http://www.seaphe.org/databases.php

11 features out of which 3 features, namely "race", "sex", "family_income", are considered as sensitive.

Like in previous datasets, we consider "sex" as sensitive feature; again the population is thus divided into male students and female students. Also, "race" is used as sensitive where the population is divided into two groups based on whether they are belonging to white sub-population. In this dataset, we consider "family_income" as sensitive. Particularly, for this sensitive feature, we divided the population into two groups based on whether the students belonged to high-income families or not.

5.2 Experimental setup

We split each dataset into 70% training set and 30% testing. As the datasets are imbalanced, we used Synthetic Minority Oversampling Technique (SMOTE[25]) over training data to generate the samples synthetically. In particular, SMOTE generates samples to balance the class label distribution in order to overcome the imbalanced distribution problem. We train original and FixOut's ensemble models on the balanced (augmented) datasets using a larger family of ML classifiers.

Note that balancing the class label distribution alone does not solve unfairness issues since we may have imbalanced distribution w.r.t. sensitive features. We do not follow any other data pre-processing treatment as we do not want to include new forms of bias in the data.

We used Scikit-learn implementation of the following classifier algorithms: AdaBoost (ADA), Bagging (BAG), Logistic Regression (LR), Random Forest (RF). We kept the default parameters of Scikit-learn documentation. In order to estimate Shapley values faster, especially in the presence of continuous features, we use K-means clustering with the number of clusters $n = 10$ to reduce feature domains; otherwise, the full domain is considered.

5.3 Experimental results

5.3.1 Process fairness assessment

Frequency of sensitive features. We start our analysis by addressing process fairness, namely, the reliance of FixOut's ensemble outputs on sensitive features. To demonstrate the ability of FixOut to reduce the reliance of classifiers on sensitive features, regardless of the choice of explanation method, we performed several experiments using LIME and SHAP explanations. The idea was to verify that by changing the explanation method (from LIME to SHAP) we can still demonstrate the impact of feature dropout on the model's reliance on sensitive features.

Figures 5.1a, 5.1b and 5.1c show the frequency of sensitive features in the top-10 most important features for both pre-trained classifiers and FixOut's ensemble models (F+SHAP, F+LIME, Fw+SHAP, Fw+LIME). For instance, the first plot in Figure 5.1a indicates the results for ADA classifiers. That is, the frequency of sensitive features in the top-10 most important features. The first group of bars indicates the frequency of the sensitive feature "sex" (the second group "telephone") in the list of top-10 most important features. Precisely, the orange bar indicates the frequency for the original model, while the pink bar indicates the frequency for FixOut +LIME. In this case, we can notice that the frequency of "sex" is lower for the ensemble model produced by F+LIME compared to the original model (8 against 17). We can observe that FixOut decreased the frequency of sensitive features in the ensemble models. We compare

[25] https://machinelearningmastery.com/threshold-moving-for-imbalanced\-classification/

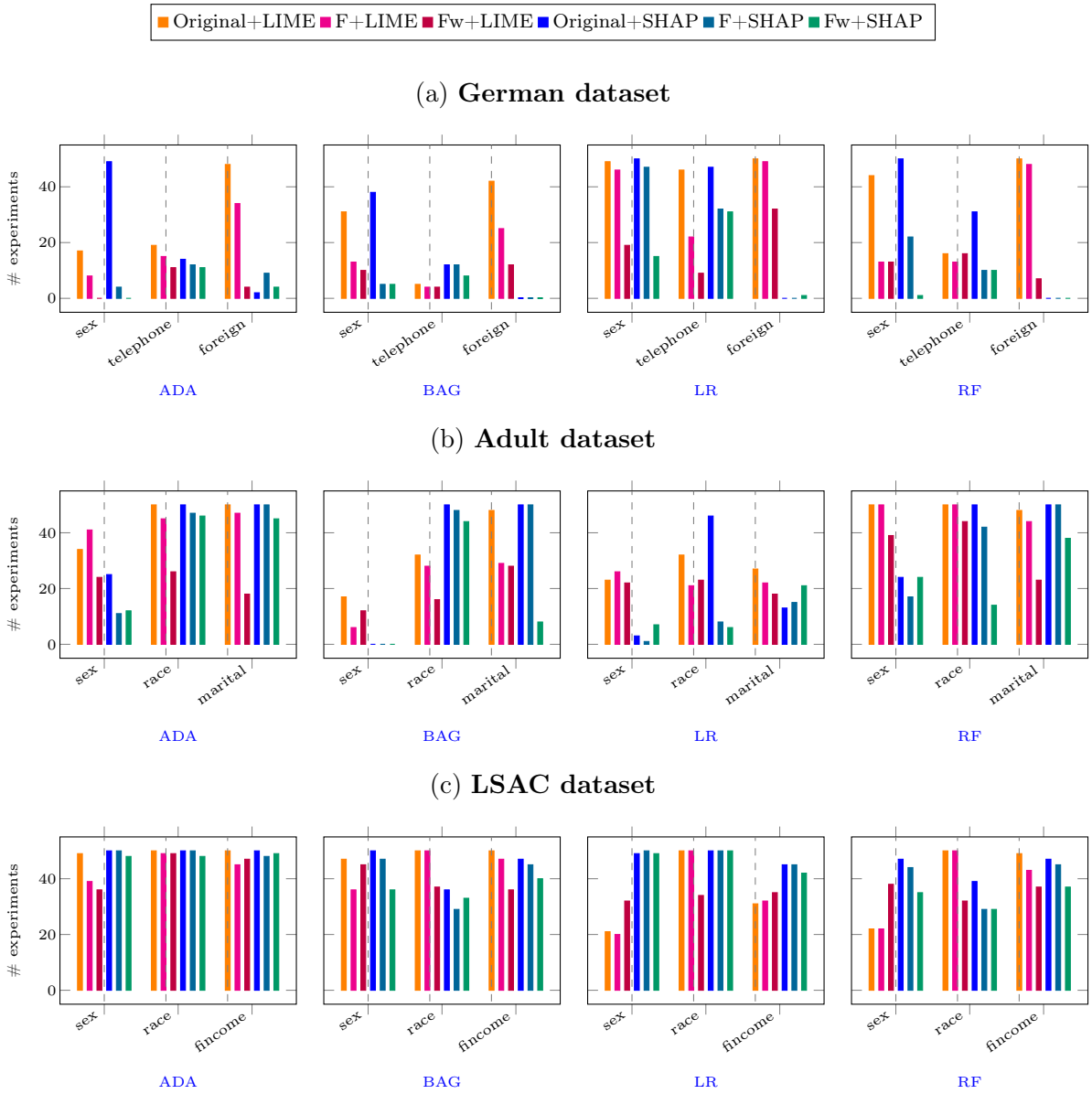


Figure 5.1: Frequency of sensitive features in the top-10 most important features for different datasets. The dashed lines separate LIME and SHAP explanations for each sensitive feature.

the original models (pre-trained classifiers with explanations obtained from LIME and SHAP, namely Original+LIME and Original+SHAP, respectively) with ensemble models produced by FixOut using different explanation methods (F+SHAP and F+LIME) and different aggregation functions (Fw+SHAP and Fw+LIME). We notice that, on the German dataset, FixOut drastically reduced the frequency of sensitive features in the top-10. One exception is for the sensitive feature "foreign" using AdaBoost.

Average contribution of sensitive features. We now analyse the value of contribution of sensitive features rather than looking at their frequency in the top-10 most important features. Table 5.2 shows the average contribution of each sensitive features throughout 50 experiments. Again, we can observe that FixOut reduces the dependency of models on sensitive features

since the average contribution of the sensitive features decreased from the original model to FixOut's ensemble models. We can also observe that it happened independently of the choice of explanation method. In other words, we noticed the same behaviour when either LIME or SHAP is applied.

Table 5.2: Average contribution of sensitive features

	Method	ADA			BAG			LR			RF		
		sex	telephone	foreign	foreign	telephone	foreign	sex	telephone	foreign	sex	telephone	foreign
German	Original+LIME	-0.13	0.12	3.84	-2.13	0.33	6.36	-13.90	10.08	25.55	-3.29	0.85	23.00
	F+LIME	-0.05	0.09	0.85	**-0.63**	0.15	1.88	-7.46	2.86	11.90	-0.55	**0.67**	7.47
	Fw+LIME	**0.00**	**0.06**	**0.02**	-0.79	**0.11**	**0.65**	**-2.00**	**1.24**	**3.28**	**-0.49**	0.69	**0.23**
	Original+SHAP	-0.68	0.10	**0.01**	-5.13	1.55	0.00	-31.20	11.59	**0.00**	-10.53	3.21	0.00
	F+SHAP	**-0.02**	**0.08**	0.04	**-0.76**	1.08	0.00	-10.20	3.52	**0.00**	**-1.87**	**0.69**	0.00
	Fw+SHAP	-0.07	**0.08**	0.13	-0.87	**0.71**	0.00	**-1.37**	**3.25**	0.06	**-1.87**	**0.69**	0.00
		marital	race	sex	marital	race	sex	marital	race	sex	marital	race	sex
Adult	Original+LIME	0.88	0.43	0.48	14.35	-1.02	**2.11**	0.49	**-0.05**	1.13	8.10	11.88	9.59
	F+LIME	0.34	0.25	0.21	2.90	**-0.40**	3.03	**-0.01**	-0.11	**0.07**	4.63	5.77	5.80
	Fw+LIME	**-0.02**	**0.13**	**0.03**	**-0.65**	-0.62	2.57	**0.01**	-0.07	0.34	**1.05**	**1.45**	**1.70**
	Original+SHAP	-3.32	0.08	0.53	98.35	0.00	**4.10**	-0.15	**0.00**	1.67	-23.29	1.29	9.25
	F+SHAP	-1.26	**0.03**	0.24	31.51	0.00	5.25	**-0.06**	**0.00**	0.06	-11.68	**0.86**	3.13
	Fw+SHAP	**-0.17**	**0.03**	**0.18**	**0.37**	0.00	4.47	-0.21	-0.02	**0.02**	**-1.73**	1.30	**0.55**
		sex	race	fincome	sex	race	fincome	sex	race	fincome	sex	race	fincome
LSAC	Original+LIME	**0.01**	0.38	0.33	2.38	-28.79	13.56	-1.35	25.30	-5.01	-0.31	-43.64	-3.48
	F+LIME	**0.02**	0.18	0.15	**1.18**	-18.02	6.31	-0.59	7.89	-1.08	-0.28	-15.18	-1.18
	Fw+LIME	**0.02**	**0.09**	**0.06**	1.72	**-0.29**	**1.81**	**-0.57**	**0.95**	**0.09**	**-0.27**	**0.38**	**-0.65**
	Original+SHAP	-0.06	0.05	-0.07	-6.59	**-0.58**	-2.43	-8.63	5.38	1.28	-3.31	-0.18	-2.70
	F+SHAP	-0.06	**0.02**	-0.05	-2.44	-0.82	-1.58	-3.50	2.74	0.41	-1.71	**-0.67**	-1.65
	Fw+SHAP	**-0.02**	**0.01**	**-0.01**	**-0.42**	-0.91	**-0.49**	**-1.03**	**1.17**	**0.29**	**-0.26**	**-0.67**	**-0.43**

5.3.2 Classification performance assessment

Table 5.3 shows the classification assessment obtained throughout the performance of the same experiment 50 times. For each dataset, we have the average accuracy, precision and recall of original models and of FixOut's ensemble models. In this analysis, we consider the following instantiations of FixOut: F+SHAP, Fw+SHAP and Fw+LIME. We did not mention F+LIME as it has equivalent results to F+SHAP.

To verify if there is a difference among the methods – i.e., the original model and the instantiations of FixOut– w.r.t. classification performance, we performed a statistical test. The idea is to determine whether at least one method behaves differently. In other words, to check if at least one of FixOut's instantiations has an impact or not in the classification performance compared to the pre-trained model. Since we have more than two methods (one original model and four FixOut's instantiations), we did the choice of the statistical test of Kruskal-Wallis.

We thus performed the Kruskal-Wallis test with 95% of confidence to assess whether the results were statistically significant among the average accuracy (also for precision and recall) of original and FixOut's ensemble models (F+SHAP, Fw+SHAP, and Fw+LIME). We found that the experiments on LSAC dataset were statistically significant, w.r.t all measures and using all classifiers. This points out that at least either the original model or one FixOut variant behaves differently. In the case of models LSAC dataset, for instance, FixOut was able to improve the

precision of models, while it maintained accuracy and recall compared to the original model. However, we did not find the same on the German dataset, where only the experiments using BAG showed statistically significant results. This points out that FixOut did not decrease nor increase classification performance, which is an expected result since FixOut decreased the dependence of models on sensitive features and at the same time maintained classification performance. In the case of experiments using BAG classifiers, ensemble classifiers produced by FixOut outperformed original models. In the case of the Adult dataset, we found that the experiments using ADA, BAG, and RF were statistically significant.

Our analysis is now based on the comparison between the classification measures of the original and FixOut's ensemble models. We notice that FixOut's ensemble models improve, or at least maintain, the accuracy and precision among the results that were statistically significant. However, we also observe that the recall decreased in the same experiments. On those that were not were statistically significant, the results indicate that FixOut's ensemble models maintain the classification measures even though we can see a slight improvement from the original model to one of FixOut's ensemble models. For instance, the accuracy obtained by LR and by Fw+SHAP on the Adult dataset.

Table 5.3: Classification assessment. Highlighted cells indicate statistical significance according to Kruskal-Wallis test with 95% of confidence.

Dataset	Method	Accuracy				Precision				Recall			
		ADA	BAG	LR	RF	ADA	BAG	LR	RF	ADA	BAG	LR	RF
German	Original	.7362	.7019	.7398	.7556	.5707	.5124	.5716	.6883	.5317	.5738	.5495	.3595
	F+SHAP	.7419	.7273	.7418	.7598	.5801	.5549	.5754	.7060	.5321	.5371	.5622	.3585
	Fw+SHAP	.7427	.7253	.7417	.7613	.5809	.5537	.5746	.7003	.5390	.5142	.5632	.3708
	Fw+LIME	.7405	.7219	.7400	.7583	.5764	.5471	.5708	.7019	.5373	.5076	.5602	.3541
Adult	Original	.8503	.8301	.6706	.8441	.6884	.6419	.3857	.7004	.6882	.6687	.5600	.6175
	F+SHAP	.8515	.8424	.6786	.8473	.6930	.6838	.3859	.7121	.6856	.6451	.5317	.6153
	Fw+SHAP	.8518	.8399	.6901	.8463	.6948	.6805	.4213	.7104	.6829	.6343	.5123	.6119
	Fw+LIME	.8512	.8388	.6713	.8470	.6927	.6771	.3922	.7108	.6838	.6343	.5398	.6156
LSAC	Original	.8398	.8544	.7526	.8513	.8986	.8846	.8548	.8771	.9016	.9413	.8330	.9473
	F+SHAP	.8331	.8620	.7440	.8553	.9044	.8898	.8596	.8838	.8850	.9448	.8136	.9436
	Fw+SHAP	.8181	.8606	.7135	.8470	.9080	.8894	.8674	.8866	.8599	.9433	.7584	.9280
	Fw+LIME	.8187	.8456	.7294	.8514	.9071	.8909	.8614	.8851	.8618	.9201	.7899	.9363

5.3.3 Fairness metrics assessment

In this section, we assess fairness using the standard metrics introduced in Chapter 1.2 in order to have a different perspective of the fairness of FixOut's ensemble models. We computed Demographic Parity (DP) and Equal Opportunity (EO) using IBM AI Fairness 360 Toolkit[26] [10]. We also considered Predictive Equality (PE) to measure the false positive differences between privileged and unprivileged groups. The metrics DP, EO, and PE give values in the interval [-1,1] where 0 indicates a perfect fair model (see the dashed line). The calculated values of fairness metrics are depicted in Figures 5.2, 5.3, and 5.4. In this analysis, we compare the original and FixOut's ensemble models based on fairness metrics for each combination of classifier and sensitive feature.

For instance, the first plot in Figure 5.2 indicates the calculated fairness metric DP of models trained on the dataset German. The calculated fairness metrics are grouped based on the type of classifier and the sensitive feature. More precisely, in this plot, the first group is "ADA-sex"

[26] https://github.com/Trusted-AI/AIF360

(x-axis) that indicates the points represent the calculated fairness notion for ADA classifiers and taking "sex" as sensitive feature. The other groups are indicated in the x-axis. Note that we calculated the fairness notions for the same ADA classifiers but taking other sensitive features into account, e.g. "ADA-foreign" indicates the results in which "foreign worker" is used to calculate the fairness metrics. Triangle violet points indicate the values for original models, while magenta points, purple points and blue points represent the values for Fw+LIME, F+SHAP and Fw+SHAP respectively. Again, the dashed line is the reference for a fair model (optimal value). Hence, points closer to the optimal value indicate better results w.r.t. one fairness metric based on one sensitive feature.

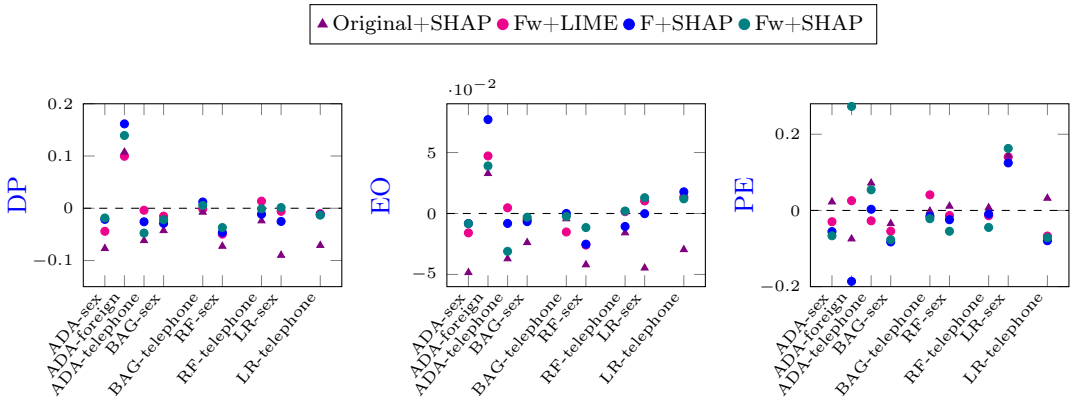


Figure 5.2: Fairness metrics for German Dataset.

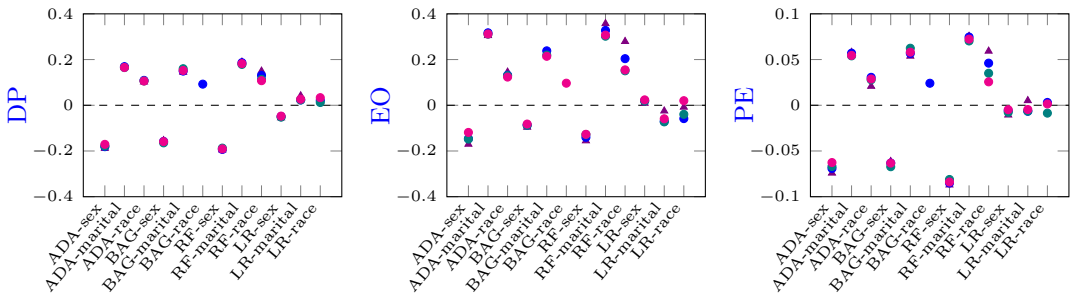


Figure 5.3: Fairness metrics for Adult Dataset.

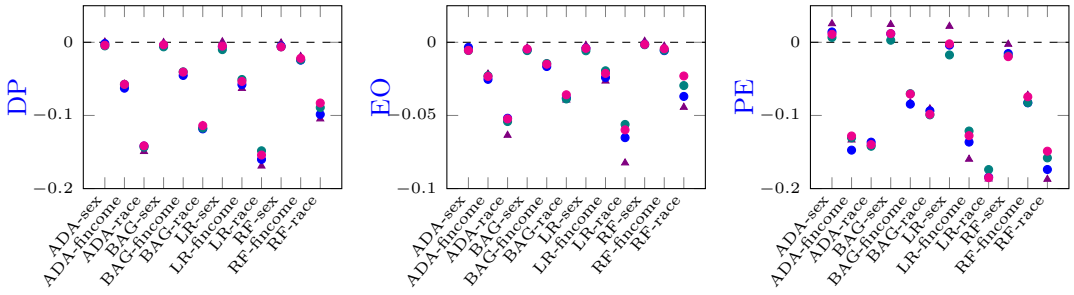


Figure 5.4: Fairness metrics for LSAC Dataset.

In this analysis, we do not compare the results with Original+LIME and F+LIME as both are equivalent to Original+LIME and F+SHAP, respectively, regarding the fairness metrics. In other words, the pairs (Original+SHAP, Original+LIME) and (F+SHAP, F+LIME) have the

same fairness metric values because explanation methods do not affect the computation of fairness metrics. In these cases, the aggregation function does not depend on the feature contributions obtained from explanations. We then simplified this analysis by ignoring Original+LIME and F+LIME.

Results for the German dataset are depicted in Figure 5.2. FixOut produces ensemble models that are fairer according to metrics DP and EO, since magenta, purble and blue points are closer to zero compared to triangle violet points (pre-trained model). However, we can not see the same behaviour for ensemble models compared to original models according to PE metric. This points out that ensemble models produced by FixOut did not improve PE, in this case, while the same ensemble models did improve EO. As satisfying PE and EO at the same time is equivalent to satisfying EqOdds (see Section 1.2.1), EqOdds is not improved in this case by FixOut either. This is not surprising, since the requirements of EqOdds are stricter than PE and EO alone, and are rarely fulfilled [60].

Figure 5.3 shows the results on fairness metrics for the Adult dataset. In this dataset, FixOut ensemble models keep values of all metrics in almost scenarios. That is, ensemble models produced by FixOut on this dataset did not improve fairness metrics overall. The exceptions are listed next. On the one hand, we can find some particular scenarios in which FixOut models were better than the original model, namely "RF-race" for all fairness metrics (DP, EO and PE), "ADA-sex" and "RF-marital" for EO. On the other hand, we can observe scenarios in which FixOut ensemble models were worse than the original model, namely "LR-marital" and "LR-race" for EO. Results depicted in Figure 5.4 concern the LSAC dataset. Again, FixOut's ensemble models maintain the fairness of models w.r.t all metrics in almost scenarios, except in the case of PE for RF on "sex". This behaviour means that FixOut at least maintain the calculated fairness metrics when it reduces the dependence on sensitive features, but it cannot ensure fairness metrics closer to the optimal value in all cases.

Again, we performed the Kruskal-Wallis test (with 95% of confidence) to assess whether the obtained results were statistically significant among the four different methods (Original+SHAP, Fw+LIME, F+SHAP, Fw+SHAP) for each combination classifier and sensitive feature w.r.t. the three fairness metrics. The results are shown in the Table 5.4. More precisely, each cell of the table indicates the result of the statistical test for a combination of classifier, fairness metric and sensitive feature. "✓" indicates that the results were statistically significant, while "×" points out the opposite. For instance, the first "✓" in the first row points out that the calculated fairness notion DP for ADA classifiers (the original model and FixOut ensemble models) w.r.t the sensitive feature "sex" of the German dataset were statistically significant. It means that the calculated fairness notion DP of FixOut ensemble models were closer to the optimal value (i.e. fairer than the original model – see the first plot of Figure 5.2) was not a random event. We noticed that some sensitive features are more impacted than others. For instance, the statistical test showed that the difference of fairness metrics computed on the sensitive feature "sex" were statistically significant. It also showed that FixOut is able to change the bias for this particular sensitive feature. However, we did not observe the same behaviour for other sensitive features. According to the same statistical test, FixOut had a significant impact when used on RF models than other models.

These results indicated that FixOut improves, or at least maintains, the fairness metrics DP and EO metrics for ensemble models. However, it also showed that there is still space for improvements. The aggregation rule should be learnt to further improve FixOut's ensemble models with respect to standard fairness metrics.

Table 5.4: Kruskal-Wallis test on fairness metrics (DP,EO, PE) with 95% confidence. ✓ indicates results among the average values of a particular fairness metric w.r.t one sensitive feature and one classifier were statistically significant, otherwise, × is used (not statistically significant). Hyphen (-) indicates that the test could not be applied for a particular sensitive feature as it does not appear in the top-10 for the original model frequently.

	Metric	German			Adult			LSAC		
		sex	telephone	foreign	marital	race	sex	sex	race	fincome
ADA	DP	✓	✓	×	×	×	✓	✓	×	×
	EO	✓	×	×	×	×	✓	✓	×	×
	PE	✓	×	×	×	×	✓	✓	×	×
BAG	DP	×	×	-	✓	-	✓	✓	×	×
	EO	×	×	-	✓	-	×	✓	×	×
	PE	×	×	-	✓	-	✓	✓	×	×
LR	DP	✓	✓	-	×	×	×	✓	✓	×
	EO	✓	✓	-	×	×	×	✓	✓	×
	PE	✓	✓	-	×	×	×	✓	✓	×
RF	DP	✓	✓	-	✓	✓	✓	✓	✓	✓
	EO	✓	×	-	✓	✓	✓	✓	✓	✓
	PE	×	×	-	✓	✓	✓	✓	×	✓

5.4 Automating the choice of FixOut's parameter

In order to assess fairness of ML models, FixOut only takes into account the first k most important features. Thus, FixOut practitioners must know beforehand a suitable value for k so that FixOut builds L with the k most important features. The system then looks at whether sensitive features are in L. An inappropriate value for k prevents FixOut of correctly detecting unfairness issues as all sensitive features may not appear in L.

For instance, if only a few features are considered to build the list of most important features (k has a low value), we may not consider any sensitive features even though one or more sensitive features have importance to the outcomes that are not negligible. Hence, FixOut will never perform the second component that reduces the importance of sensitive features. Another example in a different line is the following. Let us consider we consider instead all features when building the list of the most important features (in this case, k has a high value). Now, suppose there is one sensitive feature among all features and its importance to the outcomes is too small (i.e., negligible). Since we keep all features in the list of most important features, the sensitive feature also belongs to the list. Hence, FixOut always performs the unfairness mitigation step even though it is not needed.

In this section, we propose an algorithm that automates the choice of k, based on the statistical measure *kurtosis*. Indeed kurtosis indicates the flatness of a distribution. Let $X = (x_i)_{i \in \mathbb{N}}$ be a random variable, the *kurtosis* γ of X is defined as follows:

$$\gamma(X) = \frac{1}{n} \sum_{x_i} \left(\frac{x_i - \mu}{\sigma} \right)^4,$$

where σ is the standard deviation, μ is the mean, x_i is a sample of X, and n is the number of samples of X (i.e., the number of data instances). For instance, $\gamma(X) < 3$ indicates that X is *platykurtic* (flattened), while $\gamma(X) = 3$ defines a *mesokurtic* distribution (which is the case of normal distribution), and $\gamma(X) > 3$ indicates that the distribution is *leptokurtic* (pointed) [66,77].

Accordingly we propose the iterative algorithm called Find-K (see Algorithm 4), that selects a sub-sample of features of L based on the kurtosis measure. Find-K removes features from L by analyzing the kurtosis $\gamma(L)$ of L before and after the deletion of a subset of features. For

that, it iteratively removes features from the less important to the most important. The idea on which relies this algorithm is that once a feature is removed, the kurtosis of L changes. The algorithm stops when $|\gamma(L) - \gamma(L')| > \alpha$, where L is the original list, L' is the list obtained after removing features, and $\alpha > 0$ is a parameter.

The threshold α encodes the maximum accepted perturbation in the underlying distribution of L. The advantage of using α to determine k is that a single value of α allows us to find suitable values of k for different combinations of models and datasets.

Algorithm 4: FIND-K

Data: L: sorted list of contributions of all features (**descending** order); α: a threshold.
Result: L' a new list of contributions of subset of features.

```
L' ← L
for i ← |L| to 2 do
    L' ← L' − L'[i]
    if |γ(L) − γ(L')| > α then
        break
end
return |L'|
```

5.4.1 Selection of instances to assess fairness

We experimented different relative sizes for sampling instances. The idea is to verify whether at least one sensitive feature appears in the list of the top-10 most important features. That is, to evaluate the impact of sample sizes when assessing fairness. For that, we used the following relative sizes: 0.1%, 0.5%, 1%, 5%, and 10%. The results obtained throughout the performance of the same experiment 50 times are shown in Table 5.5. We consider four methods for selecting instances and producing explanations. These methods combine one explainer (LIME or SHAP) with a sampling method (RS or SP). We thus have in the second column: LIME+SP, LIME+RS, SHAP+SP, and SHAP+RS. For instance, the first row indicates the results w.r.t. the German dataset using LIME explanations along with the random sampling method (RS). The values in this row represent the frequency in which pre-trained models were deemed unfair using LIME and RS. Values closer to 50 indicate that the models were deemed unfair very often, while values closer to 0 indicate the opposite. In this particular row, for example, with 10% sample sizes, FixOut (using LIME and RS) considered 50 times out of 50 that the models were unfair, while with 0.5% and 1% FixOut only considered unfair the models 38 times out of 50.

For the German dataset, we did not evaluate models using 0.1% of the instances, as it takes only one instance. On the other hand, for the Adult dataset, the amount of data instances from sampling 10% of instances is large and unfeasible to produce many explanations (more than 3K) in time. We thus add an hyphen in Table 5.5 to indicate these particular cases.

Now, to illustrate the impact of the sample size and the sampling method in the assessment of fairness, we observe instead the rank of sensitive features in the list of 10 most important features ($k = 10$) obtained by each of the explainers. These results are presented in Tables 5.6, 5.7, and 5.8 for the datasets German, Adult, and LSAC, respectively.

For instance, the fist row of Table 5.6 indicates the rank of the sensitive feature "sex" in the top-10 most important features when using LIME+RS along with its average contribution in parenthesis. When 5% of data instances are sampled, "sex" appeared in the 5-th position in

Table 5.5: Number of times pre-trained models are deemed unfair. Experiments were performed by varying the number of instances selected to obtain global explanations, the sampling strategy, and the explanation method.

Dataset	**Selection**	**Sample size**				
		0.1%	0.5%	1%	5%	10%
German	*LIME+RS*	-	38	38	47	50
	LIME+SP	-	47	47	50	50
	SHAP+RS	-	50	50	50	50
	SHAP+SP	-	50	50	50	50
Adult	*LIME+RS*	46	48	49	45	-
	LIME+SP	49	49	49	49	-
	SHAP+RS	49	47	46	47	-
	SHAP+SP	47	50	49	49	-
LSAC	*LIME+RS*	50	50	50	50	50
	LIME+SP	50	50	50	50	50
	SHAP+RS	50	50	50	50	50
	SHAP+SP	50	50	50	50	50

the top-10 and it had -0.015 as contribution on average. However, with 10%, the same sensitive feature appeared in the 9-th position and it had -2.568 as contribution on average. Similarly to previous experiment, results with 0.1% are not mentioned since only one data instance would be sampled. Also, we did not mention the results of the Adult dataset for a sample size of 10% since the amount of data instances is large and unfeasible to produce many explanations.

The interesting part of these experiments is to observe how the ranking and importance of sensitive features are impacted when the sample size, explanation method and sampling method are changed. Based on these results we thus set the size as 5% for the German dataset (50 instances), 0.5% for the Adult dataset (162 instances), and 0.1% for the LSAC dataset (93 instances). These results seem to indicate that LIME is more stable than SHAP independently of the sampling method and sample size.

5.4.2 Experimental evaluation of Find-K

We evaluate the algorithm Find-K that was conceived to automatically selects the list of features which is taken into account by FixOut for assessing fairness. Find-K requires a positive value for the parameter α. Here, we performed experiments by varying the value of α from 0.5 to 3. In these experiments we analyze the impact of this parameter on the output of Find-K. In particular, we are interested in the effects of different values of α w.r.t. the size of L (i.e., the number of features kept by Find-K) and record the number of times a model is deemed unfair. We discarded values of $\alpha < 0.5$ otherwise Find-K considers all features as important.

High values of α lead to a smaller size of L, i.e., too many features are removed by Find-K. It is harder to find one sensitive feature in a very short list of important features than in a longer list. As a consequence, high values of α also lead to a low number of times in which a model is deemed unfair. We can observe this behavior on Tables 5.10 and 5.12. These tables contain the number of times that the models were deemed unfair throughout 50 repetitions of the same experiment.

A similar behavior can be observed w.r.t. the average value of k. High values of α lead to low values of k (on average). We notice this behavior in Tables 5.10 and 5.12. These tables contain

Table 5.6: Rank of sensitive features in L (and their average contribution): Effects of the variation of the sample size on the German dataset.

	Sensitive feature	Selection	Sample size 0.1%	0.5%	1%	5%	10%
German	sex	*LIME+RS*	- (-)	5 (-0.015)	4 (-0.117)	2 (-1.630)	9 (-2.568)
		LIME+SP	- (-)	6 (-0.147)	3 (-0.397)	6 (-1.610)	4 (-3.063)
		SHAP+RS	- (-)	1 (-0.345)	1 (-0.345)	0 (-3.135)	0 (-6.836)
		SHAP+SP	- (-)	0 (0.360)	0 (0.617)	0 (3.123)	0 (6.853)
	telephone	*LIME+RS*	- (-)	3 (0.003)	8 (0.095)	6 (1.509)	9 (2.891)
		LIME+SP	- (-)	3 (0.023)	5 (0.256)	6 (1.431)	7 (2.261)
		SHAP+RS	- (-)	7 (0.121)	7 (0.121)	5 (1.245)	5 (2.753)
		SHAP+SP	- (-)	2 (-0.156)	4 (-0.328)	5 (-1.180)	4 (-2.428)
	foreign	*LIME+RS*	- (-)	2 (0.041)	9 (0.076)	8 (1.078)	6 (3.294)
		LIME+SP	- (-)	1 (0.376)	0 (0.721)	0 (3.289)	0 (7.103)
		SHAP+RS	- (-)	9 (0.122)	9 (0.122)	9 (0.490)	* *
		SHAP+SP	- (-)	7 (-0.100)	9 (-0.114)	9 (-0.444)	9 (-1.011)

Table 5.7: Rank of sensitive features in L (and their average contribution): Effects of the variation of the sample size on the Adult dataset.

	Sensitive feature	Selection	Sample size 0.1%	0.5%	1%	5%	10%
Adult	marital	*LIME+RS*	9 (-0.122)	9 (-0.012)	8 (0.138)	9 (2.146)	- (-)
		LIME+SP	6 (0.042)	9 (0.367)	8 (0.244)	8 (2.746)	- (-)
		SHAP+RS	7 (-0.003)	9 (-0.239)	8 (-0.246)	7 (-0.678)	- (-)
		SHAP+SP	8 (0.030)	9 (0.093)	7 (0.296)	9 (1.731)	- (-)
	race	*LIME+RS*	9 (0.012)	6 (-0.088)	6 (-0.253)	7 (-0.093)	- (-)
		LIME+SP	9 (-0.009)	8 (-0.018)	7 (0.002)	7 (-0.002)	- (-)
		SHAP+RS	9 (0.000)	6 (0.000)	8 (0.000)	6 (0.000)	- (-)
		SHAP+SP	5 (0.000)	7 (0.000)	7 (0.001)	7 (0.000)	- (-)
	sex	*LIME+RS*	4 (-0.022)	7 (0.305)	8 (0.617)	8 (4.902)	- (-)
		LIME+SP	6 (0.153)	7 (0.823)	7 (0.484)	6 (6.925)	- (-)
		SHAP+RS	7 (0.050)	6 (0.513)	6 (1.138)	6 (2.787)	- (-)
		SHAP+SP	7 (-0.113)	6 (-0.203)	6 (-0.819)	6 (-3.387)	- (-)

Table 5.8: Rank of sensitive features in L (and their average contribution): Effects of the variation of the sample size on the LSAC dataset.

	Sensitive feature	Selection	Sample size 0.1%	0.5%	1%	5%	10%
LSAC	sex	*LIME+RS*	9 (0.045)	9 (-0.962)	9 (-1.808)	9 (-6.754)	9 (-15.629)
		LIME+SP	9 (-0.143)	9 (-0.948)	9 (-2.004)	9 (-7.074)	9 (-15.194)
		SHAP+RS	3 (-0.445)	2 (-1.986)	1 (-4.071)	1 (-18.311)	1 (-31.343)
		SHAP+SP	2 (0.477)	1 (1.827)	1 (4.943)	1 (14.738)	1 (46.189)
	race	*LIME+RS*	6 (0.735)	3 (6.309)	3 (13.561)	6 (67.047)	6 (129.130)
		LIME+SP	5 (1.222)	3 (6.827)	6 (12.202)	3 (65.198)	6 (132.644)
		SHAP+RS	4 (0.346)	3 (1.633)	2 (2.886)	3 (14.858)	3 (29.474)
		SHAP+SP	7 (-0.275)	2 (-1.385)	3 (-2.724)	3 (-15.227)	3 (-27.966)
	f_income	*LIME+RS*	8 (-0.289)	9 (-1.901)	9 (-2.342)	9 (-16.258)	7 (-45.129)
		LIME+SP	9 (-0.454)	9 (-2.250)	7 (-4.351)	7 (-25.473)	7 (-44.837)
		SHAP+RS	8 (0.119)	8 (0.487)	9 (0.812)	4 (3.913)	3 (11.642)
		SHAP+SP	8 (-0.029)	6 (-0.568)	7 (-0.989)	6 (-4.576)	6 (-6.047)

the average value of k found by Find-K. We highlighted the experiments where the average number of sensitive features found in the list obtained by FixOut was greater than or equal to 1. Again, FixOut looks for at least one sensitive feature in the list of the most important features. Then, an average smaller than 1 indicates that in most experiments the models were deemed fair. We also notice that the highlighted values are those in which the average value of k are high (see Tables 5.10 and 5.12).

We can observe that Find-K discovers on average the same value of k for each dataset. For instance, for the German dataset, an average k around 10 was found by Find-K with $\alpha = 0.5$, and incidentally the same value was used in previous empirical studies [6, 12]. For the LSAC dataset, an average k around 8 was found using Find-K with $\alpha = 0.5$, while an average k around 6 was returned by Find-K using $\alpha = 1$.

To sum up the analysis of Find-K, we can also observe that low values of α help to find a suitable value of k for users that do not have any clue of a value of k given a dataset and a classifier. More precisely, $0.5 \leq \alpha \leq 1$ allows Find-K to automatically find a suitable value of k. In addition, since LIME and SHAP have stability issues [81] that lead to different lists of feature importance (for a single type of classifier trained on the same dataset), it may be interesting to considering the threshold α to learn the parameter k. Indeed, one single value of α allows Find-K to find a suitable value of k from distinct lists of feature importance. This provides an answer to question Q4 introduced at the beginning of this chapter –how to reduce human intervention in FixOut usage on tabular data.

5.5 Discussion and perspectives

In this chapter, we have instantiated FixOut to address and tackle process fairness on models trained on tabular data. The empirical study was performed on 3 real tabular datasets. It showed that FixOut drastically reduces a model's reliance on sensitive features (i.e., improved process fairness) without compromising accuracy and precision, regardless of the choice of explanation method.

It also showed a significant decrease in both the feature contributions and the frequencies of

Table 5.9: Average value of k for LR classifiers (number of times models are deemed unfair): Effects of the variation of α over the number of features kept by FIND-K.

Dat.	Selection	LR					
		α					
		0.5	1	1.5	2	2.5	3
German	*LIME+RS*	9.9 (47)	8.7 (42)	6.8 (31)	5.5 (23)	3.5 (13)	2.6 (8)
	LIME+SP	9.7 (50)	8.2 (49)	6.1 (43)	4.5 (39)	2.1 (29)	1.3 (24)
	SHAP+RS	10.0 (50)	9.9 (50)	9.5 (50)	7.4 (47)	5.7 (45)	3.8 (43)
	SHAP+SP	10.0 (50)	9.8 (50)	8.9 (50)	7.0 (49)	5.4 (45)	4.0 (43)
Adult	*LIME+RS*	10.0 (46)	10.0 (46)	10.0 (46)	9.9 (46)	9.9 (44)	9.8 (44)
	LIME+SP	10.0 (49)	10.0 (49)	10.0 (49)	10.0 (49)	9.9 (49)	9.8 (49)
	SHAP+RS	10.0 (48)	10.0 (48)	10.0 (48)	10.0 (48)	9.8 (48)	9.7 (45)
	SHAP+SP	10.0 (50)	9.9 (50)	9.8 (50)	9.7 (50)	9.6 (49)	9.3 (49)
LSAC	*LIME+RS*	8.6 (47)	6.4 (28)	4.0 (10)	2.5 (6)	1.8 (1)	1.3 (0)
	LIME+SP	7.7 (35)	5.1 (22)	3.1 (14)	1.7 (4)	1.1 (0)	1.0 (0)
	SHAP+RS	8.5 (48)	6.6 (42)	4.5 (30)	3.2 (22)	2.5 (18)	2.0 (12)
	SHAP+SP	8.3 (50)	6.1 (45)	4.4 (36)	3.2 (26)	2.3 (19)	1.7 (14)

Table 5.10: Average value of k for BAG classifiers (number of times models are deemed unfair): Effects of the variation of α over the number of features kept by FIND-K.

Data.	Selection	BAG					
		α					
		0.5	1	1.5	2	2.5	3
German	*LIME+RS*	10.0 (33)	9.58 (31)	8.18 (24)	6.38 (18)	4.26 (8)	2.82 (2)
	LIME+SP	10.0 (41)	10.0 (41)	9.74 (41)	9.08 (40)	7.04 (30)	4.86 (21)
	SHAP+RS	9.92 (37)	9.80 (37)	8.78 (33)	7.16 (26)	5.86 (22)	4.86 (19)
	SHAP+SP	10.0 (45)	9.78 (45)	9.12 (43)	7.72 (36)	6.58 (31)	5.50 (26)
Adult	*LIME+RS*	10.0 (49)	10.0 (49)	10.0 (49)	9.58 (49)	8.32 (45)	6.82 (40)
	LIME+SP	10.0 (50)	10.0 (50)	10.0 (50)	9.64 (50)	8.52 (50)	7.32 (50)
	SHAP+RS	10.0 (49)	10.0 (49)	10.0 (49)	9.62 (49)	8.30 (45)	7.08 (40)
	SHAP+SP	10.0 (50)	10.0 (50)	8.84 (50)	5.84 (39)	2.32 (11)	1.32 (2)
LSAC	*LIME+RS*	7.74 (50)	5.64 (49)	4.04 (43)	2.68 (32)	1.98 (24)	1.56 (21)
	LIME+SP	9.72 (50)	8.76 (50)	7.86 (50)	7.00 (50)	5.96 (49)	4.84 (44)
	SHAP+RS	8.34 (50)	5.98 (32)	4.46 (24)	3.04 (17)	2.56 (14)	2.08 (11)
	SHAP+SP	8.70 (49)	6.88 (44)	5.10 (33)	3.76 (26)	2.82 (14)	2.22 (11)

Table 5.11: Average value of k for RF classifiers (number of times models are deemed unfair): Effects of the variation of α over the number of features kept by Find-K.

Data.	Selection	RF					
		α					
		0.5	1	1.5	2	2.5	3
German	*LIME+RS*	9.90 (49)	8.30 (43)	5.18 (25)	2.72 (8)	1.54 (2)	1.18 (2)
	LIME+SP	10.0 (50)	9.98 (50)	9.74 (50)	8.54 (50)	6.96 (47)	5.46 (46)
	SHAP+RS	9.92 (50)	8.98 (48)	6.46 (39)	4.52 (29)	2.74 (23)	2.04 (18)
	SHAP+SP	9.86 (50)	8.70 (47)	5.64 (33)	3.68 (27)	2.28 (22)	1.42 (17)
Adult	*LIME+RS*	9.76 (50)	8.38 (50)	7.00 (48)	5.48 (38)	4.44 (18)	3.64 (6)
	LIME+SP	9.30 (50)	7.80 (50)	6.76 (50)	5.80 (48)	5.00 (40)	4.32 (27)
	SHAP+RS	10.0 (50)	9.96 (50)	9.30 (50)	8.12 (50)	6.48 (46)	5.14 (41)
	SHAP+SP	10.0 (50)	9.98 (50)	9.38 (50)	8.02 (48)	6.26 (47)	4.94 (38)
LSAC	*LIME+RS*	6.46 (49)	4.04 (38)	2.02 (16)	1.46 (10)	1.10 (6)	1.02 (6)
	LIME+SP	8.92 (50)	7.30 (49)	5.38 (48)	3.66 (48)	2.66 (48)	1.96 (48)
	SHAP+RS	7.80 (47)	5.80 (40)	3.88 (21)	2.48 (13)	1.76 (7)	1.40 (3)
	SHAP+SP	8.18 (49)	5.78 (37)	4.04 (23)	2.86 (13)	2.10 (8)	1.70 (4)

Table 5.12: Average value of k for ADA classifiers (number of times models are deemed unfair): Effects of the variation of α over the number of features kept by Find-K.

Data.	Selection	ADA					
		α					
		0.5	1	1.5	2	2.5	3
German	*LIME+RS*	10 (48)	10 (48)	9.76 (48)	9.4 (46)	9.2 (45)	9.04 (44)
	LIME+SP	10.0 (50)	10.0 (50)	10.0 (50)	10.0 (50)	10.0 (50)	10.0 (50)
	SHAP+RS	10.0 (48)	8.78 (43)	6.44 (31)	4.28 (23)	3.24 (15)	2.90 (13)
	SHAP+SP	9.98 (47)	8.68 (42)	6.78 (34)	5.82 (30)	4.92 (24)	4.04 (20)
Adult	*LIME+RS*	10.0 (50)	10.0 (50)	8.22 (49)	5.54 (35)	3.40 (13)	2.20 (2)
	LIME+SP	10.0 (50)	9.90 (50)	7.98 (49)	5.74 (43)	3.84 (19)	2.48 (4)
	SHAP+RS	10.0 (50)	9.02 (50)	6.62 (49)	4.76 (48)	3.28 (47)	2.38 (47)
	SHAP+SP	10.0 (50)	9.16 (49)	7.22 (49)	5.36 (49)	4.06 (47)	2.82 (47)
LSAC	*LIME+RS*	6.98 (44)	4.52 (32)	3.02 (20)	1.92 (11)	1.28 (5)	1.12 (4)
	LIME+SP	7.08 (46)	5.06 (34)	3.44 (22)	2.22 (11)	1.78 (9)	1.28 (4)
	SHAP+RS	8.68 (50)	6.52 (42)	4.78 (30)	3.34 (21)	2.08 (8)	1.52 (6)
	SHAP+SP	8.84 (50)	7.08 (47)	5.42 (39)	3.84 (26)	2.68 (12)	1.90 (7)

sensitive features in the top-10 of most important features when using SHAP or LIME explanations. In fact, we observed a similar behaviour between FixOut with SHAP and FixOut with LIME with respect to classification performances and process fairness (**Q1**). We also evaluated the impact of two different aggregation functions on FixOut's output models. The comparison of the aggregation rules employed, namely, the simple average and the weighted average rules, revealed that the latter rule had a beneficial impact on process fairness (**Q2**). Even though FixOut was designed to improve process fairness, we also assessed FixOut empirically using well known fairness metrics. Our results showed overall improvements with respect to some fairness metrics and sensitive features. However, FixOut did not guarantee improvements in all cases (**Q3**) and this asks for a deeper study in this direction. It also asks for evaluating FixOut on extensions of LIME and SHAP, e.g. BayesLIME and BayesSHAP. Finally, other ways to learn the aggregation function can be studied in the future.

Chapter 6

Mitigating unfairness on textual data: FixOut's extensions

Contents

6.1 FixOut applied to models trained on textual data 72
6.2 Experimental setup 75
6.3 Experimental results 75
6.3.1 Process fairness assessment 75
6.3.2 Classification performance assessment 76
6.4 FixOut for Neural Networks 77
6.5 Discussion and perspectives 79

FixOut has been initially proposed to address algorithmic unfairness on ML models trained on tabular data. The empirical studies have shown that the framework improve process fairness without compromising classification performance. However, fairness issues have also been found in models trained on textual data, ranging from simple models (e.g. linear regression) to complex and opaque ones (e.g. deep neural networks). For instance, Papakyriakopoulos et al. [69] trained word embeddings on texts from Wikipedia articles and from political social media (tweets and Facebook comments) that are written in gendered languages[27]. They then investigated the diffusion of bias when these embeddings are used in ML models. In addition, they proposed approaches to reduce bias in embeddings based on gender, ethnicity and sexual orientation. In spite of eliminating bias in word embeddings, the authors state that it might include other types of bias.

In this chapter, we extend FixOut to be applied on models for textual data and we apply this FixOut's extension in the task of classifying tweets as hate speech or not [5]. We take advantage of FixOut that address algorithmic unfairness through unawareness, and extend it to classification scenarios dealing with textual data. The hypothesis here is that FixOut *is able to improve process fairness not only in tabular data but also in textual data.* It is addressed by investigating the following research question

Q5 Does FixOut reduce model dependence on sensitive words of models trained on textual data?

[27]The grammar of gendered languages require that speakers mark gender in some parts of speech, for instance in nouns and/or adjectives (e.g. German and French).

We take advantage of the FixOut proposal that address algorithmic unfairness through unawareness, and extend it to other settings, namely classification scenarios dealing with textual data.

6.1 FixOut applied to models trained on textual data

In this section, we demonstrate the adaptability of the FixOut framework to another data type, namely textual data. In this setting, a data instance corresponds to a text and features to words, i.e., the number of features is equal to the vocabulary size. To drop a word, we remove each occurrence of the corresponding word in the text. Here words that are semantically equivalent are considered different.

Let M be a classifier for which we want to reduce the impact of words from a given set S, in the classifier's outcomes:

$$S = \{s_i \mid 1 \leqslant i \leqslant n\}.$$

As in tabular data, FixOut decreases the contribution of the words in S to the classifiers outcomes by applying *word dropout* and by building an ensemble of the $n + 1$ classifiers:

- one classifier M_i, for $1 \leqslant i \leqslant n$, trained by ignoring the word s_i, and
- a classifier $M_{[n+1]}$ trained by ignoring all words in S.

However, when n is large, this method becomes inefficient, for two reasons. The first one is that if we consider a sensitive word $s_i \in S$, only two classifiers are ignoring it: M_i and $M_{[n+1]}$, i.e., $n - 1$ out of the $n + 1$ classifiers are taking the feature s_i into account. Hence, the dropout of s_i may then become less significant when n is large. The second reason is a complexity concern: the more models there are, the more memory and computation time the ensemble takes.

Textual dataset. The dataset used to evaluate this version contains tweets written in two language variant: African-American English and Standard-American English [24]. Classifiers trained on that dataset were reported to be biased against tweets written in African-American English. For instance, some words that are considered offensive in Standard-American English are used in familiar interactions in African-American English, e.g. between close friends, and they do not indicate offensive discussions.

As language variant should not be a criteria for classifying a tweet as hate speech or not, FixOut was extended in order to reduce the dependence of classifiers on certain words. To do so, feature dropout was adapted to word dropout. However, once the number of words to be removed increases, FixOut becomes less effective. In order to overcome this issue, instead of ignoring a single word, words are grouped in order to perform a "bag of word dropout", i.e., the framework drops several words. The contribution of words using bag of words dropout (grouping words) is lower than word dropout (without grouping words).

A simple example of word dropout. Table 6.1 shows results for a random forest model, which classifies a text as hate speech or not. We select $n = 3$ words for which we want to reduce the contribution. The list of words used here was inspired by the analysis performed in [23]. Each row of Table 6.1 is dedicated to one word s_i. It indicates the position of s_i and the average contribution in the list of most important words for the original model and the FixOut ensemble. For instance, in the first row of Table 6.1 we observe that *"niggah"* is placed 14th in the initial position, i.e. in the list of most important words of the original model, and 30th in the

ranking of the FixOut ensemble model. To obtain the contribution (importance) of each word to model's outcome, we apply an explanation method once the model is trained as we did in the case of tabular data. Since the explanation methods LIME and SHAP have a lack of stability, we train the same type of classifier 50 times. For each execution, we asked for explanations (along with a sampling technique, such as RS and SP). We then calculate the average contribution of a word throughout 50 executions. We can check that the rank of contribution (importance) of each dropped word decreases thanks to the FixOut ensemble. More precisely, given a word s_i, we evaluate the gain $drank(\cdot)$ of s_i by calculating the difference between the rank obtained (r'_i) and the starting rank (r_i), normalised by the smaller of the two as follows

$$drank(s_i) = \frac{r_i - r'}{min(r'_i, r_i)}.$$

For instance, the gain rank of *"niggah"* indicated in the first row of Table 6.1 (column "Diff Rank" is calculated as $(14 - 30)/\min(14, 30) = 1.14$. Gaining a few ranks on the highest features in the ranking allows us to have significant scores, whereas they will be close to 0 for ranks that are very far away.

Table 6.1: Process fairness assessment on a hate speech classifier (RF) using SHAP+RS, selecting 3 words

	Original model		FixOut Ensemble		**Diff**
Word	**Rank**	**Contrib.**	**Rank**	**Contrib.**	**Rank**
niggah	14	0.176	30	0.043	1.14
nigger	12	0.213	29	0.045	1.41
nig	7	0.276	55	0.021	6.85

Bag of words dropout. When n increases and more than 3 words are added, the method seems to become less effective, as shown in Table 6.2 (column "Without grouping"), which describes the rank and contribution of each one of the selected words whose importance should be reduced. Let us turn our attention to the interest of grouping words. Instead of ignoring a single word or feature, the classifier can drop many words at the same time, then reducing the size of the ensemble. We proceeded in this way and the performance of FixOut was improved as it can be observed in Table 6.2.

Table 6.2: Process fairness assessment on a hate speech classifier (RF) with SHAP+RS, selecting 7 words

	Without grouping		With grouping		**Diff**
Word	**Rank**	**Contrib.**	**Rank**	**Contrib.**	**Rank**
niggah	18	0.149	23	0.03	0.27
nigger	15	0.164	21	0.031	0.40
nigguh	22	0.13	83	0.008	2.77
nig	12	0.202	65	0.011	4.41
nicca	22	0.107	39	0.018	0.77
nigga	20	0.125	12	0.067	-0.66
white	25	0.087	36	0.018	0.44

We also experimented FixOut with simple textual classifiers. In particular, we implemented a model used in an experiment carried out by Davidson et al. [23], whose goal is to classify

tweets as *hate speech* or not. We focus on the *hate speech* dataset [24], which more precisely labels a tweet as *offensive*, *hate speech*, or *neither*. To stay within a two-class problem, we merge *offensive language* and *hate speech* classes, to finally deal with two classes, namely *non-offensive* or *offensive*.

However, the resulting dataset appears to be quite unbalanced, including 4163 non-offensive instances and 20620 offensive ones. Thus we randomly select only 4163 instances from the offensive group. Moreover, we follow the same representation and pre-processing steps as in [23]. We stem each word in order to reduce and to gather similar words, e.g., *"vehicle"* and *"vehicles"* are both transformed into *"vehicl"*. The classifier is based on a TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer followed by a logistic regression. In addition, we ran experiments with a bagging ensemble, a random forest, and AdaBoost, instead of the logistic regression.

Then we apply the first step of FixOut to build a global explanation. This is performed with the textual adaptation of LIME and SHAP. We compare both random sampling (RS) and submodular pick (SP), for these two explainers. Unlike the previous tabular classifiers, the model deals with more than 2000 features, i.e., as many as the vocabulary size. For example, Table 6.3 shows the first 10 most important words obtained after 50 experiments run with a logistic regression and LIME explanations, respectively with random sampling and submodular pick.

Table 6.3: Mean contribution of selected words for trained LR model. Contributions were obtained from LIME explanations throughout 50 experiments.

Random sampling (RS)		
Rank	**Word**	**Contrib.**
1	*faggot*	0.632
2	*fag*	0.625
3	*bitch*	0.622
4	*niggah*	0.620
5	*cunt*	0.613
6	*pussi*	0.608
7	*nigger*	0.597
8	*hoe*	0.580
9	*nigguh*	0.522
10	*dyke*	0.473

Submodular pick (SP)		
Rank	**Word**	**Contrib.**
1	*niggah*	0.596
2	*cunt*	0.579
3	*bitch*	0.574
4	*fag*	0.573
5	*faggot*	0.573
6	*nigger*	0.572
7	*pussi*	0.547
8	*hoe*	0.524
9	*nigguh*	0.505
10	*retard*	0.414

At first glance, we can observe that important words for hate speech classification are insults and swear words, which makes sense. However, it should be noticed that such a classification may also be related to a particular context. For example, the use of words such as *"n*gger"* in a conversation may also be related to a familiar interaction between two very close friends, and thus not at all indicate an *offensive* discussion.

This shows that the definition of a *sensitive word* is not straightforward in such a case, and that we must find words that are responsible for a bias. Then, in order to complete this experiment, we manually select sensitive features. Accordingly, the objective of the next subsection is to demonstrate that FixOut is able to decrease the contribution of pre-selected words thanks to a FixOut ensemble and by grouping words. For example, considering the groups of words presented in Figure 6.1, the goal of this ensemble is to reduce the contribution of these groups of words.

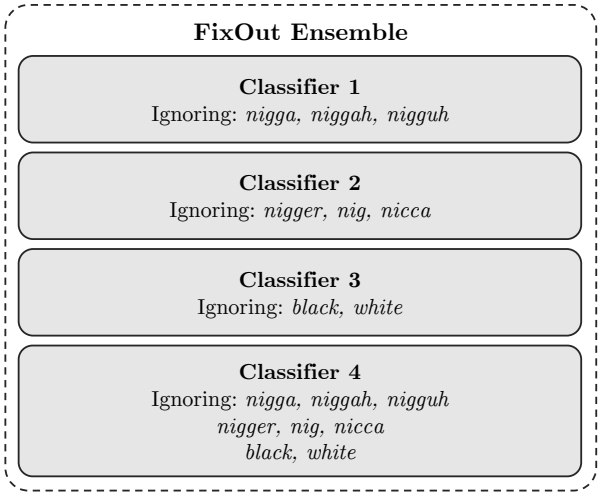


Figure 6.1: Illustration of textual classifiers used in the ensemble produced by FixOut.

6.2 Experimental setup

For each experiment, we train the same type of classifier we used before, i.e. either LR, RF, ADA, or BAG, plus four classifiers corresponding to the related FixOut ensembles (the instantiations of FixOut), which should be less dependent to sensitive words as defined above. Next, we use LIME and SHAP explanations, with both random sampling and submodular pick (i.e., LIME+RS, LIME+SP, SHAP+RS, and SHAP+SP). For each explanation, we assess the contribution of all the features. We then sort all the features w.r.t. their average contribution, select the 500 most important words, and then check their rank. Words which are not in this list are therefore ranked beyond 500 and their contribution is marked as " - " to indicate that it is not significant.

6.3 Experimental results

6.3.1 Process fairness assessment

Table 6.4 shows the results obtained with random sampling, including ranking difference. LIME explanations systematically show a decrease in the contribution and the importance rank of selected words. We even observe that the contribution is often nearly divided by 2, although the contribution of the whole vocabulary tends to decrease a bit in the ensemble explanation. By contrast, SHAP explanations show an increase in the ranking of the word *nigga*, although its contribution decreases. The contribution of all other selected words is decreasing in general, even if this tendency seems to be more instable than in the case of LIME. Table 6.4 also presents the results obtained with submodular pick, and shows results similar to random sampling with better results with logistic regression (+1.22 in average with LIME and +1.90 with SHAP). These latter results seem to indicate that FixOut with LIME explanations is more stable and consistently improves process fairness, i.e., always decrease the contribution of sensitive features. However, FixOut with SHAP seems to provide greater improvements; see, e.g., "nigguh" and "nig" in the case of LR (diff. rank 22.81 and 54.56 with submodular pick and 21.73 and 40.67 with random sampling, resp.). It is noteworthy that for SHAP "nigga" always worsens.

Table 6.4: Process fairness assessment on textual data, global explanations with submodular pick and random sampling, respectively.

		Original model		FixOut Ensemble		Diff
	Word	**Rank**	**Contrib.**	**Rank**	**Contrib.**	**Rank**
LR, LIME+SP	*niggah*	1	0.596	12	0.275	11
	nigger	6	0.572	13	0.274	1.16
	nigguh	9	0.505	14	0.269	0.55
	nig	13	0.315	20	0.138	0.54
	nicca	15	0.289	23	0.12	0.53
	nigga	18	0.24	24	0.119	0.33
	white	23	0.178	45	0.096	0.96
	black	145	0.047	323	0.028	1.23
RF, LIME+SP	*niggah*	7	0.517	12	0.257	0.71
	nigger	9	0.476	15	0.23	0.67
	nigguh	13	0.339	17	0.194	0.31
	nig	10	0.445	16	0.204	0.60
	nicca	16	0.265	20	0.121	0.25
	nigga	17	0.235	23	0.112	0.35
	white	23	0.127	34	0.07	0.48
	black	>500	~ 0	>500	~ 0	0.00
ADA, LIME+SP	*niggah*	2	0.167	4	0.083	1.00
	nigger	7	0.052	10	0.026	0.43
	nigguh	5	0.144	6	0.073	0.20
	nig	18	0.014	24	0.006	0.33
	nicca	17	0.015	23	0.007	0.35
	nigga	4	0.166	5	0.083	0.25
	white	23	0.011	26	0.005	0.13
	black	113	0.0	196	0.0	0.73
BAG, LIME+SP	*niggah*	5	0.645	15	0.293	2.00
	nigger	11	0.58	16	0.29	0.45
	nigguh	13	0.497	18	0.233	0.38
	nig	1	0.659	10	0.336	9.00
	nicca	15	0.387	19	0.195	0.27
	nigga	19	0.285	21	0.139	0.11
	white	21	0.183	24	0.083	0.14
	black	420	0.004	>500	~ 0	0.19

		Original model		FixOut Ensemble		Diff
	Word	**Rank**	**Contrib.**	**Rank**	**Contrib.**	**Rank**
LR, SHAP+SP	*niggah*	16	0.179	75	0.015	3.69
	nigger	11	0.239	23	0.034	1.09
	nigguh	21	0.123	>500	~ 0	22.81
	nig	9	0.249	>500	~ 0	54.56
	nicca	22	0.12	46	0.02	1,09
	nigga	19	0.159	14	0.053	-0.36
	white	20	0.14	50	0.019	1,50
	black	>500	~ 0	>500	~ 0	0.00
RF, SHAP+SP	*niggah*	14	0.176	23	0.03	0.64
	nigger	12	0.213	21	0.031	0.75
	nigguh	22	0.13	83	0.008	2.77
	nig	7	0.276	65	0.011	8.29
	nicca	21	0.138	39	0.018	0.86
	nigga	18	0.155	12	0.067	-0.50
	white	26	0.085	36	0.018	0.38
	black	>500	~ 0	>500	~ 0	0.00
ADA, SHAP+SP	*niggah*	5	0.05	11	0.007	1.20
	nigger	9	0.018	13	0.005	0.44
	nigguh	6	0.039	25	0.002	3.17
	nig	18	0.009	86	0.001	3.78
	nicca	34	0.006	40	0.001	0.18
	nigga	4	0.07	3	0.031	-0.33
	white	32	0.007	30	0.002	-0.07
	black	>500	~ 0	>500	~ 0	0.00
BAG, SHAP+SP	*niggah*	15	0.202	24	0.021	0.60
	nigger	12	0.28	17	0.038	0.42
	nigguh	19	0.166	44	0.012	1.32
	nig	2	0.508	14	0.044	6.00
	nicca	14	0.209	21	0.025	0.50
	nigga	17	0.195	11	0.078	-0.55
	white	25	0.074	72	0.008	1.88
	black	>500	~ 0	>500	~ 0	0.00

6.3.2 Classification performance assessment

We also performed a classification performance evaluation based on accuracy, precision and recall. Like in the tabular data scenario, the goal is to compare pre-trained models against ensemble models produced by FixOut when detecting hate speech. The results of this evaluation are presented in Table 6.5. For each row in the table, we have the calculated performance of original models and FixOut ensemble models for a particular type of classifier. We can observe that, overall, FixOut ensemble models maintained the classification performance over different classification metrics. The last row of the table shows the mean difference between the classification performance of the original model and FixOut's ensemble model. For instance, the biggest difference is found for the recall metric (-0.008) and the smallest for the precision (-0.002). This indicates that despite the reduction of the model's dependence on particular words, FixOut produced ensemble models that do have similar classification performance. That is, this FixOut's extension is able to find a good compromise between classification performance and fairness in the case of hate speech detection.

	Word	Original model Rank	Original model Contrib.	FixOut Ensemble Rank	FixOut Ensemble Contrib.	Diff Rank
LR, LIME+RS	*niggah*	4	0.62	12	0.277	2.00
	nigger	7	0.597	10	0.29	0.43
	nigguh	9	0.522	14	0.264	0.56
	nig	13	0.328	20	0.15	0.54
	nicca	14	0.316	22	0.134	0.57
	nigga	18	0.237	27	0.113	0.50
	white	23	0.172	35	0.104	0.52
	black	131	0.05	316	0.028	1.41
RF, LIME+RS	*niggah*	7	0.593	10	0.289	0.43
	nigger	9	0.497	14	0.236	0.56
	nigguh	13	0.358	17	0.198	0.31
	nig	10	0.46	16	0.222	0.60
	nicca	14	0.286	20	0.132	0.43
	nigga	18	0.23	23	0.112	0.28
	white	23	0.119	33	0.066	0.43
	black	>500	~ 0	>500	~ 0	0.00
ADA, LIME+RS	*niggah*	3	0.166	5	0.082	0.67
	nigger	7	0.052	11	0.026	0.57
	nigguh	5	0.144	6	0.074	0.20
	nig	18	0.014	24	0.006	0.33
	nicca	17	0.015	23	0.007	0.35
	nigga	4	0.166	4	0.083	0.00
	white	22	0.011	26	0.005	0.18
	black	115	0.0	192	0.0	0.67
BAG, LIME+RS	*niggah*	6	0.687	13	0.32	1.17
	nigger	10	0.629	15	0.306	0.50
	nigguh	13	0.535	17	0.248	0.31
	nig	2	0.7	10	0.353	4.00
	nicca	15	0.395	19	0.2	0.27
	nigga	19	0.284	21	0.139	0.11
	white	22	0.158	25	0.074	0.14
	black	>500	~ 0	>500	~ 0	0,00

	Word	Original model Rank	Original model Contrib.	FixOut Ensemble Rank	FixOut Ensemble Contrib.	Diff Rank
LR, SHAP+RS	*niggah*	16	0.194	68	0.016	3.25
	nigger	7	0.273	18	0.042	1.57
	nigguh	22	0.126	>500	~ 0	21.73
	nig	12	0.223	>500	~ 0	40.67
	nicca	21	0.138	37	0.024	0.76
	nigga	19	0.179	13	0.061	-0.46
	white	20	0.177	53	0.019	1.65
	black	>500	~ 0	>500	~ 0	0.00
RF, SHAP+RS	*niggah*	14	0.188	22	0.03	0.57
	nigger	12	0.219	21	0.031	0.75
	nigguh	23	0.132	84	0.008	2.65
	nig	10	0.272	68	0.011	5.80
	nicca	21	0.154	36	0.021	0.71
	nigga	17	0.17	12	0.075	-0.42
	white	25	0.11	38	0.019	0.52
	black	>500	~ 0	>500	~ 0	0.00
ADA, SHAP+RS	*niggah*	5	0.048	11	0.007	1.20
	nigger	9	0.019	13	0.005	0.44
	nigguh	6	0.039	24	0.002	3.00
	nig	20	0.008	80	0.001	3.00
	nicca	26	0.006	37	0.001	0.42
	nigga	4	0.072	3	0.031	-0.33
	white	22	0.008	28	0.002	0.27
	black	>500	~ 0	>500	~ 0	0.00
BAG, SHAP+RS	*niggah*	17	0.203	26	0.02	0.53
	nigger	13	0.257	16	0.039	0.23
	nigguh	19	0.165	46	0.012	1.42
	nig	2	0.442	14	0.044	6.00
	nicca	14	0.217	21	0.026	0.50
	nigga	16	0.206	11	0.082	-0.45
	white	24	0.091	53	0.009	1.21
	black	>500	~ 0	>500	~ 0	0.00

Table 6.5: Classification assessment. Comparison between the original model and the ensemble produced by FixOut.

	Accuracy		Precision		Recall	
	Original	Ensemble	Original	Ensemble	Original	Ensemble
LR	0.928	0.921	0.93	0.922	0.925	0.919
RF	0.953	0.951	0.961	0.960	0.945	0.942
ADA	0.952	0.949	0.967	0.970	0.937	0.926
BAG	0.947	0.941	0.954	0.951	0.939	0.929
Mean diff.	-0.005		-0.002		-0.008	

6.4 FixOut for Neural Networks

The framework FixOut was also adapted to be model-specific; particularly, to be used on top of neural networks. However, LIME and SHAP suffer from the fact they take a lot of time to generate explanations for classifiers based on neural networks. In addition, neural networks also take a lot of time to be re-trained. So, the main idea behind this version of FixOut (so-called FixOut-NN) is to have an extended framework which is efficient for neural networks. The second component of FixOut builds a pool of classifiers in order to obtain an ensemble. FixOut has to re-train classifiers which is time consuming specially if classifiers are neural networks. Unlike

FixOut, FixOut-NN employs the same pre-trained classifier (neural network) in the ensemble (see Figure 6.2b). To do so, FixOut-NN uses word embeddings to represent textual data. The pre-trained classifier is modified multiple times in order to build the ensemble classifier thanks to different filters (in the pre-processing step) that manipulate word embeddings (i.e., obtaining multiple representations). Hence, there is no need for re-training the pre-trained classifier in the pool of classifiers, which takes much less time and memory. In addition, the pool of filters is used to keep the differentiability of the whole ensemble. Here, the filters are responsible for applying the equivalent word dropout approach to embeddings. That is, remove the parts of the embedding that correspond to the tokens representing the sensitive words. This is an important point since only replacing sub-models by neural networks within the ensemble does not make it differentiable. The reason for this is that we need to isolate the process of obtaining embeddings out of the ensemble model. Thus, the input of the ensemble is the embedding, and then the differentiable part covers from the input to the output of the ensemble. This difference is depicted in Figures 6.2. The ensemble model produced by the traditional version of FixOut for textual data keeps the process of obtaining embeddings inside the ensemble model (red part of Figure 6.2a), which is a non-differentiable part. On the other hand, FixOut-NN takes the embedding as input, not the text (the process to obtain embeddings is out of the ensemble model), and only keeps the filters along with a neural network. Since the obtained ensemble from FixOut-NN is differentiable (as well as the pre-trained neural network), a model-specific explanation method can be applied. FixOut-NN employs, in this case, PathExplain (Section 3.6), instead of LIME or SHAP.

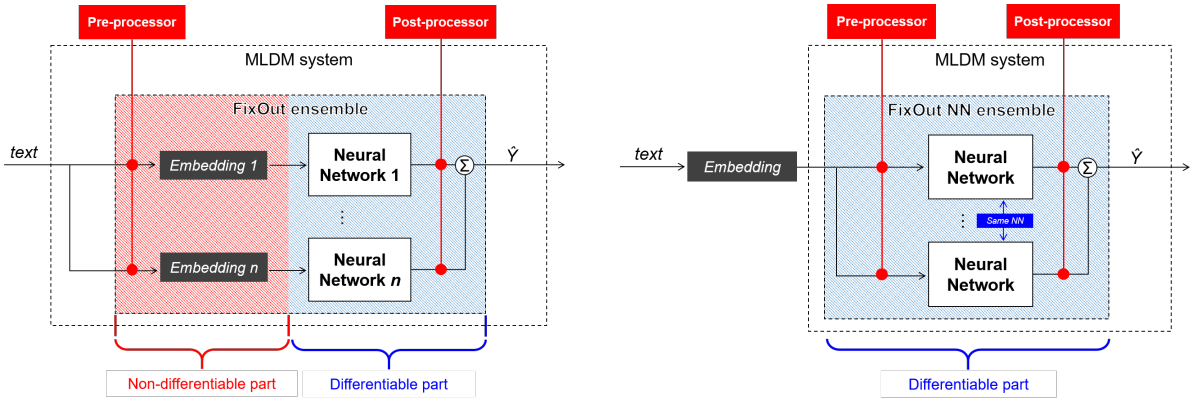


Figure 6.2: Scheme of (a) an ensemble produced by FixOut and (b) another ensemble produced by the model-specific extension of FixOut to neural networks, FixOut-NN. FixOut contains a non-differential part (in red) which does prevent from obtaining explanations using PathExplain. However, FixOut-NN does not contain a non-differential part and it is fully differentiable.

Experiments were performed with this version of FixOut with PathExplain on the same dataset of tweets in which the goal is to detect whether a tweet is offensive. The obtained results are shown in Tables 6.6 and 6.7. The comparison of contribution and rank of words between the original and FixOut-NN can be observed in Table 6.6. For instance, FixOut-NN reduced the contribution of the word *niggah* from 0.109 (original model) to 0.071 (ensemble model produced by FixOut-NN). The classification assessment is presented in Table 6.6. Again, the classification performance were not affected since we only observe slightly differences between the original and

FixOut-NN ensembles. More precisely, we obtain a mean difference of 0.008 for accuracy, 0.006 for precision, and 0.005 for recall.

	Original model		FixOut-NN	
Word	**Rank**	**Contrib.**	**Rank**	**Contrib.**
niggah	18	0.109	29	0.071
nigger	3	0.408	9	0.272
nigguh	46	0.045	56	0.035
nig	37	0.061	50	0.041
nicca	6	0.303	13	0.202
nigga	5	0.328	12	0.219
white	19	0.100	27	0.079
black	107	-0.018	132	-0.012

Table 6.6: Fairness assessment of models (original and FixOut-NN) trained on textual data using explanations from PathExplain.

	Original	**FixOut-NN**
Accuracy	0.947	0.939
Precision	0.951	0.945
Recall	0.946	0.941

Table 6.7: Classification assessment of the original and the ensemble models produced by FixOut-NN.

6.5 Discussion and perspectives

In this chapter, we revisited the framework FixOut that was proposed to render fairer classification models when applied to tabular data. We explored the adaptability of FixOut's framework to classification models on textual data. For that, we adapted the notion of feature dropout to bag of words, followed by the ensemble strategy. Our empirical results showed the feasibility of this idea when rendering models fairer. Furthermore, the comparison of FixOut's workflow with LIME and SHAP explanations indicated that, even though SHAP provides drastic improvements in some cases, FixOut with LIME explanations is more stable, and consistently results in fairer models; this is not the case when using SHAP explanations. Submodular pick gives better average results than random sampling and the best gain is obtained for Logistic regression for both LIME and SHAP (and for both SP and RS).

This contribution opens several avenues for future work. For instance, in the case of textual data it remains to automate the choice of the words that FixOut should take into account in the dropout approach based on bags of words. FixOut-NN can be used with other gradient based explanation methods. Furthermore, we are currently experimenting with different combinations of fairness processors, such as combining pre-, in-, and post-processing fairness interventions.

Conclusion and perspectives

In this thesis, we tackled the tension between fairness and classification performance (the fairness-accuracy trade-off) in the context of binary classification. We proposed methods that output ensemble models in order to find a suitable compromise between fairness and classification performance.

Summary

One of the main challenges of this thesis was to propose model-agnostic approaches for unfairness mitigation that take advantage of ensemble-based methods. The key idea of the solutions described herein was found by combining two different types of fairness interventions: pre-processing and post-processing. On top of this challenge, we noticed the lack of approaches for improving process fairness, a specific fairness notion. The generic framework FixOut was thus proposed to improve this fairness notion without compromising classification performance and at the same time keep the model-agnostic characteristic. The general idea of FixOut was presented in Chapter 4.

In order to cope with the difficulty of assessing process fairness, i.e., estimating the impact of sensitive features throughout a ML pipeline, we took advantage of explanation methods. We thus investigated how to use explanations to assess and improve process fairness. We did an empirical by employing two well-known post-hoc explanation methods, namely: LIME and SHAP. The empirical analysis was possible thanks to two main instantiations of FixOut, each one for a specific data type.

In Chapter 5, the instantiation of FixOut for tabular data and the empirical analysis is presented accordingly. However, a recent body of work raised concerns in approaches based on explanation methods that use feature importance indices to assess fairness since they conceal other forms of unfairness. This motivated us to deepen the thorough analysis of FixOut to evaluate the outcomes of FixOut with respect to well-known fairness notions such as DP,EO,PE. Our results show improvements in some notions with a very few exceptions. We also applied several aggregation rules in the post-processing step of FixOut.

In Chapter 6, the instantiations of FixOut for textual data with an empirical analysis on the task of hate speech classification is presented. The goal of the empirical analysis was to demonstrate the adaptability of FixOut in rendering classification models fairer when applied to textual data. For that, we adapted the notion of feature dropout to the bag of words dropout. We noticed that removing a single word is not effective to reduce the dependence of the model on that word since we have several words to be considered as sensitive. Essentially, the idea of bag of words dropout is to remove a set of words rather than one word, so that we have less classifiers to be trained and also less classifiers that take a sensitive word in to account. The empirical analysis then showed that bag of words dropout is computationally efficient and

effective compared to training separately one model after dropping out one sensitive word.

Pros. One of the main advantages of FixOut is its adaptability. The framework is model-agnostic and can be used on top of any classifier. We performed experiments using four different classifiers (ADA, LR, RF, and BAG). Another advantage of our proposed framework is that it improves a notion of fairness that is not well-studied in the literature and does not suffer from the problem of mathematical reductionism. Moreover, in specific contexts, FixOut can also improve well-know fairness notions.

Cons. FixOut is not designed to improve fairness of a specific and well-known fairness notion, even though it is able to improve one or more fairness notions (w.r.t. one or more sensitive features) as a side effect of improving process fairness. More precisely, in its current version, it does not allow users to enforce a specific notion of fairness other than process fairness. For instance, it is not possible to perform FixOut to improve only EO along with process fairness. Also, high-correlated features with sensitive features have to be treated before-hand by ML practitioners since FixOut does not automatically identify those features. In addition, regarding the FixOut version for textual data, determining which words and the group of words that will be dropped should be also studied and informed by users before running FixOut.

Limitations. As FixOut relies on explanation methods, some of its limitations come from this dependence. For instance, post-hoc explanation methods, such as LIME and SHAP, assign importance to features based on how they impact an individual prediction, however they do not explain why some features are important and others are not. As a result, FixOut is not able to explain why a sensitive feature appear in the top-k most important features. More over, explainers are vulnerable to adversarial attacks that conceal unfairness. Accordingly, FixOut is also vulnerable to adversarial attacks, as are explanation methods based on perturbations.

Perspectives

In each contribution chapter (Chapters 5 and 6), we discussed how each contribution could be extended in specific contexts. Here, we point out more general and transversal perspectives for our future work.

- **Fairness notions.** A body of work on fairness notions has recently tried to recommend the suitable fairness notion to be used given a specific problem. As FixOut is limited to process fairness, a future work may be followed in the direction of extending the framework to automate part of the choice of the fairness notions and improve the recommended notion along with process fairness.

  In a different research line, the scientific community is interested in causal-based methods in order to study unfairness issues in ML models. These methods tackle the problem of fairness through causal inference and they present the following advantages: (1) they can do a proper analysis of discrimination in the presence of statistical anomalies, e.g. Simpson's paradox, (2) they provide an interpretation of the relationship between features and the outcomes, and (3) they are able to detail the dependence of outcomes to sensitive features which help to explain why some features are more important than others. To sum up, incorporating causal notions of fairness into FixOut could help to increase the quality of explainability and to break down the discrimination before and after applying the fairness interventions by our proposed framework.

- **Fairness processors.** Even though FixOut already combine more than one fairness interventions, i.e., it is a hybrid-processing fairness processor, there is still room to incorporate

other fairness processors into our proposed framework. An immediate approach is to combine some of the fairness pre-processors (mentioned in Section 2.1) into the pre-processing phase of FixOut. Similarly, another immediate approach is to incorporate other fairness post-processors (from Section 2.3) into the post-processing phase of FixOut in order to enforce even more fairness.

- **Explanations and fairness.** Despite the increased interest in explainability in ML and a plethora of recent papers proposing explanators, explanations are still considered fragile. Relying only on explanations to assess and mitigate unfairness might hide unfairness, particularly in the case of feature importance explanations. In order to render FixOut robust, an interesting direction might be the use specific explanations based on the domain or prefer likelihood explanations rather than feature importance explanations, which is a feasible objective since the advent of more robust and stable explanators (Section 3.5).

- **Beyond batch data and classification.** Recently, with the advent of explanation methods for different data types (e.g. TimeSHAP [11]), an interesting direction would be extending the idea of feature dropout with ensemble approach to tackle fairness on models for sequential data. To the best of our knowledge, until here, this idea is being applied to tackle only unfairness in binary classification task. Another perspective is to extend this approach to different tasks, e.g. regression and clustering.

Appendix A

Correlation analysis

For each dataset described in Chapter 5, we performed a correlation analysis among pairs of features. The goal was to identify if there exist sensitive features highly-correlated with non-sensitive features. We thus calculated the Pearson correlation coefficient for all pairs of features. The correlation matrices are depicted in Figures A.1, A.2, and A.3. Considering a threshold of 0.75, we did not find any pair (sensitive, non-sensitive) of features that has a *absolute* correlation value ≥ 0.75 in all datasets considered.

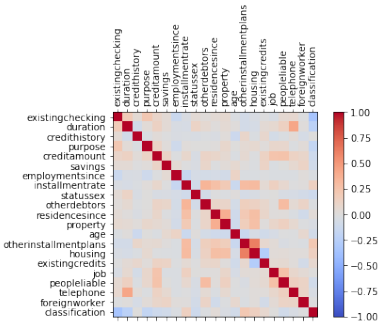


Figure A.1: German

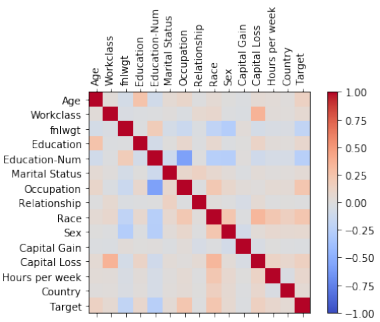


Figure A.2: Adult

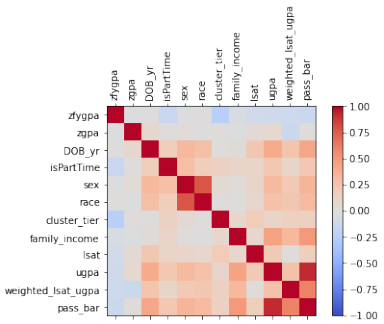


Figure A.3: LSAC

Appendix B

Further experiments I

FixOut applied to models trained on other datasets

HMDA. The *Home Mortgage Disclosure Act*[28] aims to help identifying possible discriminatory lending practices. This public data about home mortgage contains information about the applicant (demographic information), the lender (name, regulator), the property (type of property, owner occupancy, census tract), and the loan (loan amount, type of loan, loan purpose). Here, the goal is to predict whether a loan is "high-priced", and the features that are considered sensitive are "sex", "race", and "ethnicity".

Default. This dataset is also available on the UCI repository[29]. The goal is to predict the probability of default payments using data from Taiwanese credit card users, e.g., credit limit, gender, education, marital status, history of payment, bill and payment amounts. We consider as sensitive features in this dataset: "sex" and "marriage".

Table B.1: Average accuracy assessment, where FixOut stands for the ensemble model built by our proposed framework. Numbers in parentheses indicate standard deviation. No accuracy values are reported on the HMDA dataset for logistic regression, and on the Default dataset for random forest and logistic regression, since in each of these cases the original model was deemed fair.

		ADA	BAG	RF	LR
HMDA	Original	.879 (.001)	**.883** (.001)	.882 (.001)	.878 (.001)
	FixOut	.880 (.001)	.884 (.000)	.884 (.000)	-
Default	Original	**.817** (.003)	.804 (.003)	.807 (.003)	.779 (.004)
	FixOut	.817 (.003)	.812 (.002)	-	-

[28] https://www.consumerfinance.gov/data-research/hmda/

[29] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

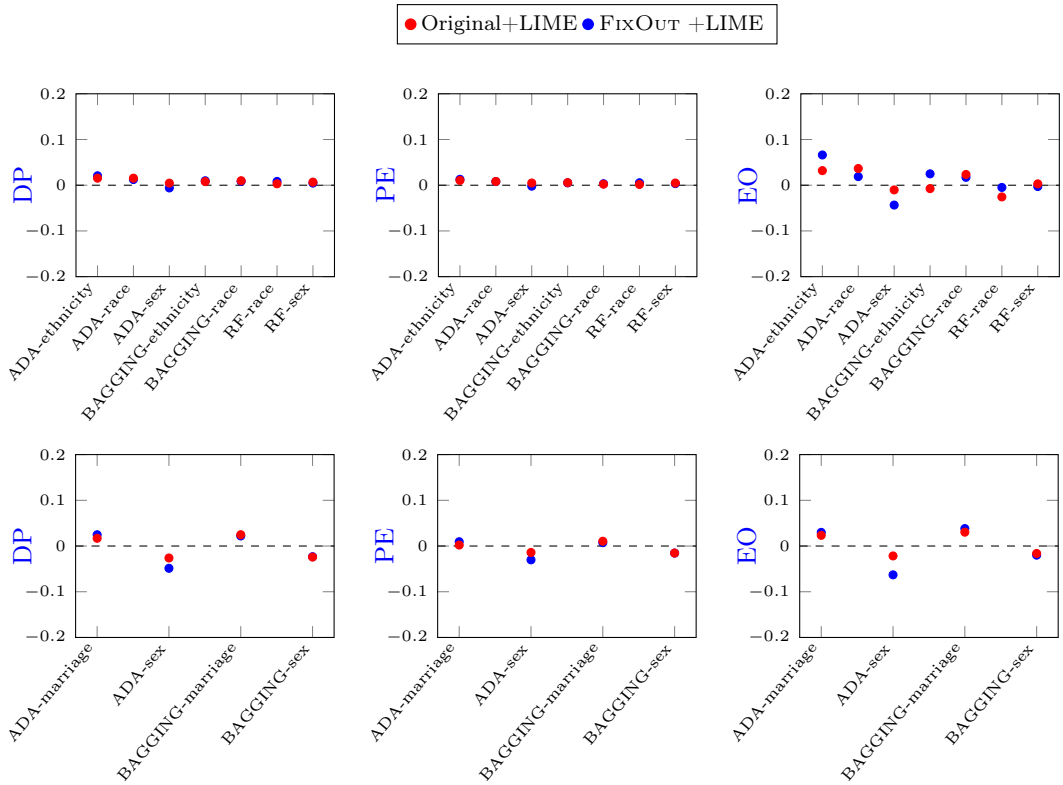


Figure B.1: Fairness metrics for the HMDA dataset (first and second lines) and the Default dataset (third and fourth lines). For both datasets, lesser original models were deemed unfair, namely, ADA, BAG and RF on HMDA, and ADA and BAG on Default. Even though these models were deemed unfair by FixOut, most of the fairness metrics actually indicate a rather fair behaviour by the original and FixOut's ensemble models.

Table B.2: Fairness assessment of BAG on HMDA dataset.

Original (SHAP)

Feature	Contrib.
loan_product_t	131.970375
intro_rate_period	55.185479
race	-35.760327
sex	12.466373
applicant_score_t	12.222446
debt_to_income	6.906509
applicant_above_62	6.140392
loan_amount	5.489757
loan_to_value_ratio	5.421845
income	5.189954

Ensemble (SHAP)

Feature	Contrib.
loan_product	152.401853
intro_rate_period	78.719783
applicant_score_d	29.645684
loan_to_value_ratio	21.613061
loan_amount	8.680493
applicant_score	7.692752
debt_to_income	7.539624
loan_term	-7.261209
income	5.350675
applicant_above_62	4.415154

Figure B.2: Sensitive features in the top-10.

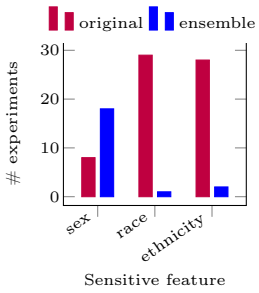


Table B.3: Fairness assessment of ADA on Default dataset.

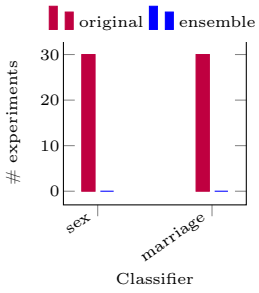
Original (SHAP)

Feature	Contrib.
PAY_0	-0.970901
PAY_AMT2	-0.533411
EDUCATION	0.40984
PAY_AMT3	-0.388931
PAY_5	-0.183335
PAY_6	-0.138245
MARRIAGE	-0.056857
PAY_2	-0.048885
PAY_3	-0.028558
SEX	-0.001967

Ensemble (SHAP)

Feature	Contrib.
PAY_0	-0.711716
PAY_AMT1	-0.482153
PAY_AMT3	-0.347062
EDUCATION	0.290634
AGE	0.229882
PAY_6	-0.188315
PAY_3	-0.122395
PAY_5	-0.103442
PAY_2	-0.08513
PAY_AMT2	0.07256

Figure B.3: Sensitive features in the top-10.



Appendix C

Further experiments II

FixOut and other fairness processors

To show how fairness notions are used to assess fairness and to illustrate FixOut in different scenarios, two distinct datasets were used, namely *communities and crimes* and *Compas*. For each one of them, the most common fairness notions are computed in four scenarios: baseline model (logistic regression (LR) including all the features in the dataset), baseline model after Reweighing (pre-processor presented in 2.1.1), baseline model along with Threshold Optimizer (post-processor presented in 2.3.1), and FixOut 's ensembles (hybrid-processor of pre-processing along with post-processing) using LR. This allows to show how feature dropout through process fairness produces an ensemble classifier with a good trade-off between fairness and classification accuracy.

Datasets

Compas. This dataset contains information from Broward County, Florida, initially compiled by ProPublica [7] and the goal is to predict the two-year violent recidivism. That is, whether a convicted individual would commit a violent crime in the following two years (1) or not (0). Only black and white defendants who were assigned *Compas* risk scores within 30 days of their arrest are kept for analysis [7] leading to 5915 individuals in total. We consider *race* as sensitive feature in the first setting and *gender* in the second. Each categorical feature is transformed to a set of binary features leading to 11 features in total.

Communities and crimes. This dataset[30] includes information relevant to per capita violent crime rates in several communities in the United States and the goal is to predict this crime rate. The dataset includes a total number of 123 numerical features and 1994 instances. 22 features have been dropped as they contain more than 80% missing values. The label *violent crime rate* has been transformed into a binary feature by thresholding[31] where 1 corresponds to high violent rate and 0 corresponds to low violent rate. To assess fairness, we consider two different settings depending on the sensitive feature at hand. First, the *communities racial makeup* is considered as the sensitive feature thus, two groups are created, namely: whites (communities with high rate of whites) and non-whites (communities with high rate of non-whites[32]). Second, the *communities rate of divorced female* is used as sensitive feature where we divide the samples

[30] https://archive.ics.uci.edu/ml/datasets/communities+and+crime

[31] The mean value of the violent crime rate in the dataset is used as threshold.

[32] Blacks, Asians, Indians and Hispanics are grouped into a single group called non-whites.

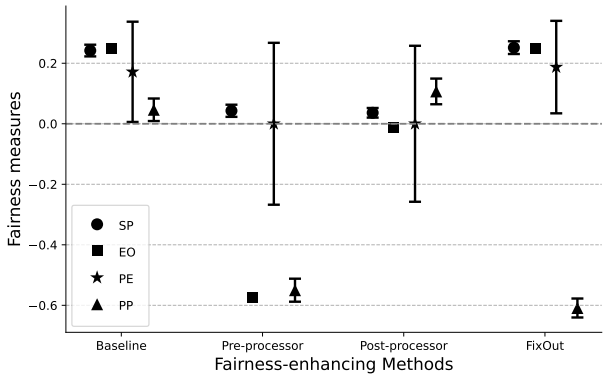


Figure C.1: Fairness assessment for the Compas dataset with *race* as a sensitive feature.

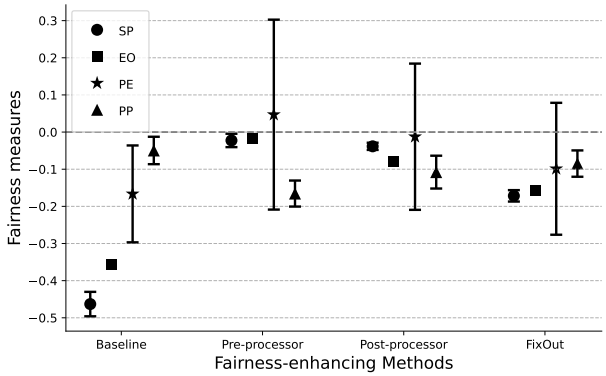


Figure C.2: Fairness assessment for the Compas dataset with *sex* as a sensitive feature.

into two sub-populations based on whether the rate of divorced females in a community is high (1) or low (0)[33].

We do not see any decrease w.r.t feature contribution and ranking. Note that classification accuracy is almost the same for FixOut, the original model and the pre-processor, but it decreased when the post-processor was applied (see Figure C.5).

Calibration analysis

Now, we present a fairness assessment based on the fairness notion calibration. We binned the predicted scores in calibration in 10 bins and we then calculate the bin-centers for each bin as shown in Table C.1 and C.2. For instance, the results show discrimination against communities with high rate of non-whites in the first setting and against communities with high rate of divorced females in the second setting for all fairness notions except for some of the calibration results corresponding to the bold-faced rates[34] presented in Table C.1.

[33]The mean value of the divorced female rate in the dataset is used as threshold.
[34]We consider here a maximum difference of 0.01 as insignificant.

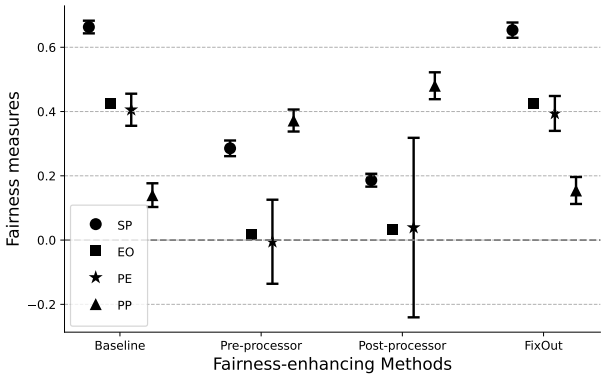


Figure C.3: Fairness assessment for the communities and crimes dataset with *race* as a sensitive feature.

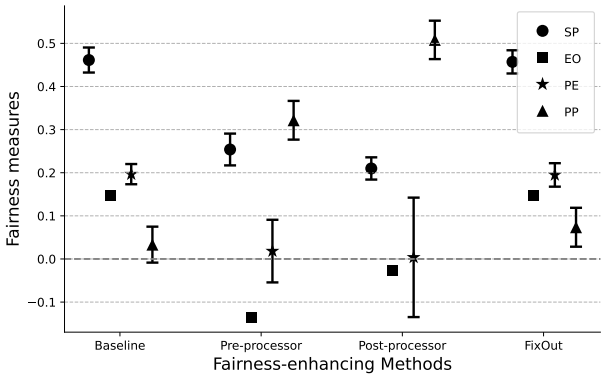


Figure C.4: Fairness assessment for the communities and crimes dataset with *divorced female rate* as a sensitive feature.

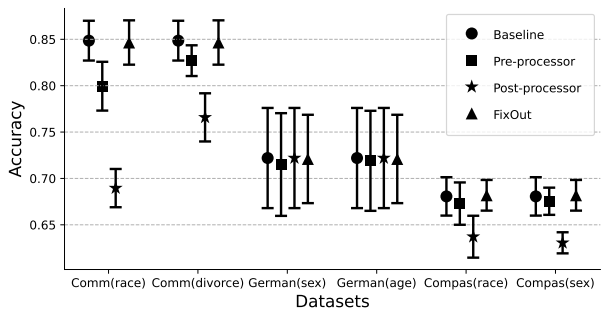


Figure C.5: Accuracy of all datasets after applying the different fairness-enhancing methods.

Table C.1: Calibration obtained from experiments on the *communities and crimes* dataset.

Sensitive feature	Baseline (LR)										FixOut									
	bin centers										*bin centers*									
	.13	.21	.30	.38	.46	.54	.63	.71	.80	.88	.16	.26	.33	.40	.47	.54	.61	.66	.75	.82
Race	.08	**.20**	.29	.36	.45	.52	.60	.67	.75	.81	.08	**.20**	.29	.36	.44	.52	.60	.68	.75	.82
	.14	**.21**	.30	.41	.48	.56	.64	.71	.81	.90	.14	**.21**	.31	.42	.48	.56	.63	.72	.80	.91
Divorced	.20	.24	.31	.39	**.46**	**.53**	**.60**	**.68**	.76	.87	.20	.24	.31	.39	**.45**	.52	**.60**	.68	.76	.87
female rate	.01	.12	.28	.35	**.46**	**.54**	**.60**	**.67**	.72	.83	.06	.17	.28	.36	**.44**	.54	**.60**	.66	.73	.83

Table C.2: Calibration obtained from experiments on the Compas dataset.

Sensitive feature	Baseline (LR)										FixOut									
	bin centers										*bin centers*									
	.10	.19	.28	.37	.46	.54	.63	.72	.81	.90	.10	.19	.28	.37	.46	.54	.63	.72	.81	.90
Race	**.16**	**.21**	.27	**.37**	**.45**	**.53**	**.62**	**.71**	**.79**	**.86**	**.17**	.21	.27	**.36**	**.45**	**.53**	**.62**	.70	**.78**	**.86**
	.17	**.22**	.29	**.37**	**.46**	**.54**	**.63**	**.72**	**.79**	**.87**	**.17**	.23	.29	**.37**	**.46**	**.54**	**.63**	.72	**.79**	**.87**
Gender	**.16**	.20	**.27**	**.37**	**.44**	**.53**	**.62**	.70	**.79**	.85	**.17**	**.22**	**.28**	**.37**	**.45**	**.54**	**.63**	.72	**.79**	**.87**
	.17	.22	**.28**	**.37**	**.45**	**.54**	**.63**	.72	**.79**	.87	**.17**	**.21**	**.27**	**.37**	**.45**	**.53**	**.62**	.70	**.78**	**.86**

Appendix D

List of publications

International conferences

- G. Alves, A. Amblard, F. Bernier, M. Couceiro, A. Napoli. **Reducing Unintended Bias of ML Models on Tabular and Textual Data**. In 8th IEEE International Conference on Data Science and Advanced Analytics, pages 1–10. Porto, Portugal. (2021)
- G. Alves, V. Bhargava, M. Couceiro, A. Napoli. **Making ML models fairer through explanations: the case of LimeOut**. In International Conference on Analysis of Images, Social Networks and Texts, pages 3–18. Moscow, Russia (2020)

Preprints

- G. Alves, V. Bhargava, F. Bernier, M. Couceiro, A. Napoli. **FixOut: an ensemble approach to fairer models**. ⟨hal-03033181⟩. (2020)

Others

During the doctoral studies, additional research works were done in other topics. These originated the following papers:

- G. Alves, M. Couceiro, A. Napoli. **Sélection de mesures de similarité pour les données catégorielles**. In 20ème édition de la conférence Extraction et Gestion des Connaissances, Brussels, Belgium. (2020)
- G. Alves, M. Couceiro, A. Napoli. **Towards a Constrained Clustering Algorithm Selection**. In Société Francophone de Classification Actes des 26èmes Rencontres, Nancy, France. (2019)

Glossary

ADA AdaBoost. 56

AI Artificial Intelligence. 1

BAG Bagging. 56

GDPR General Data Protection Regulation. 1

LIME Local Interpretable Model-agnostic Explanations. 4

LR Logistic Regression. 56

ML Machine Learning. 1

MLDM ML-based Decision Making. 1

RF Random Forest. 56

RS Random Sampling. 45

SHAP SHapley Additive exPlanations. 4

SP Submodular Pick sampling. 37, 45

Notation

A^* sensitive feature. 13

A feature. 14

B desired number of explanations. 37

C set of class labels. 49

D dataset (set of data instances). 38

E explanation method. 44

I array of feature importance. 37

K length of explanation. 35

L list of feature importance. 45

M classification model. 13

S score. 14

V selected instances. 37

W explanation matrix. 37

Y actual outcome. 11

Ω complexity of explanation. 33

α alpha. 63

γ kurtosis. 62

$\mathcal{A}^*$ list of sensitive features. 44

$\mathcal{L}$ loss function. 33

$\mathcal{M}$ maximum coalition size. 36

ϕ feature importance. 33

π kernel. 33

d distance function. 17

f prediction function. 33

g linear (interpretable) model. 33

h_x transformation function. 33

k number of features considered. 45

x data instance. 17

z interpretable representation of x. 33

BA Balanced Accuracy. 13

BN Balance for Negative Class. 15

BP Balance for Positive Class. 15

BR Base Rate Prevalence. 13

CAE Conditional Use Accuracy Equality. 16

DI Disparate Impact. 14

DP Demographic parity. 13

EA Equal Accuracy. 16

EO Equal Opportunity. 15

EqOdds Equalized Odds. 15

FDR False Predictive Value. 13

FN False Negative. 13

FNR False Negative Rate. 13

FOR False Omission Rate. 13

FP False Positive. 13

FPR False Positive Rate. 13

FTA Fairness Through Awareness. 17

FTU Fairness Through Unawareness. 17

NPV Negative Predictive Value. 13

OA Overall Accuracy. 13

PE Predictive Equality. 15

PP Predictive Parity. 16

PPV Posivite Predictive Value. 13

TN True Negative. 13

TNR True Negative Rate. 13

TP True Positive. 13

TPR True Positive Rate. 13

Résumé étendu

Les biais jouent un rôle important pour les modèles d'apprentissage automatique (*machine learning* – ML), puisqu'ils sont conçus pour avoir un certain biais qui les guide dans leurs tâches cibles. Cependant, les modèles ML ont divers biais qui peuvent provenir de sources multiples et certains peuvent ne pas être souhaités [61]. Si les biais ont un impact inattendu sur la société, en particulier contre les minorités, qui devraient plutôt être protégées, alors cela caractérise un problème d'équité [64]. Par exemple, des discriminations fondées sur le genre et l'origine ethnique ont été détectées sur des platesformes en ligne pour les demandes d'emploi [43]. Un biais basé sur les variantes linguistiques a également été constaté dans les approches de détection des propos abusifs, les messages rédigés en anglais afro-américain obtenant des scores de toxicité plus élevés que ceux rédigés en anglais américain standard [23].

Ce type de biais cachés dans les systèmes de prise de décision basés sur le ML (ML-*based decision making* – MLDM) peut automatiser les décisions injustes, ce qui dévoile le besoin d'approches qui découvrent et suppriment (ou réduisent) les biais inattendus. L'évaluation de l'équité et l'atténuation de l'iniquité sont les deux principales tâches qui ont motivé la croissance du domaine de recherche appelé *algorithmic fairness* (équité algorithmique) [8, 64].

Pour atténuer ou supprimer l'iniquité, les approches appliquent généralement des interventions en matière d'équité selon des étapes spécifiques. Récemment, les recherches sur l'équité algorithmique ont été consacrées à l'exploration de combinaisons de différentes interventions en matière d'équité, ce qui est désigné dans cette thèse par le *traitement hybride de l'équité*. Une fois que nous essayons d'atténuer l'iniquité, une tension entre l'équité et la performance apparaît, connue comme le compromis équité/précision (*fairness-performance trade-off*) [53].

Dans cette thèse, nous proposons des méthodes basées sur les ensembles pour trouver un bon compromis entre l'équité et la performance de classification des modèles ML, en particulier les modèles pour la classification binaire. De plus, ces méthodes produisent des classificateurs d'ensemble grâce à une combinaison d'interventions sur l'équité, ce qui caractérise les approches de traitement hybride de l'équité.

Nous proposons FixOut (*FaIrness through eXplanations and feature dropOut*)[35], un framework centré sur l'humain et agnostique par rapport au modèle qui réduit la dépendance des classificateurs aux caractéristiques sensibles sans compromettre leurs performances de classification. Il reçoit en entrée un classificateur (modèle pré-entraîné), un ensemble de données, un ensemble de caractéristiques sensibles et une méthode d'explication, et il produit un nouveau classificateur qui dépend moins des caractéristiques sensibles. Pour évaluer la dépendance d'un modèle pré-entraîné donné aux caractéristiques sensibles, FixOut utilise des explications pour estimer la contribution des caractéristiques aux résultats des modèles. S'il est démontré que les caractéristiques sensibles contribuent globalement aux résultats des modèles, alors le modèle est considéré comme injuste. Dans ce cas, un pool de classificateurs plus justes est construit, qui

[35] https://fixout.loria.fr/

sont ensuite agrégés pour obtenir un classificateur d'ensemble.

Les principales contributions de cette thèse sont :

1. **Introduction de FixOut.** Nous proposons un framework générique qui aborde le problème de la recherche d'un bon compromis entre les performances de classification et l'équité du processus, appelé FixOut. Nous réalisons une étude empirique afin d'évaluer le framework en utilisant différents classificateurs. Cette contribution peut être décomposée en plusieurs parties.

2. **Automatisation des paramètres du framework.** Nous avons proposé un algorithme qui sélectionne automatiquement le nombre de caractéristiques que FixOut prend en compte lors de l'évaluation de l'équité. Cet algorithme offre un outil utile dans les scénarios de présence de données à haute dimensionnalité (nombre de caractéristiques).

3. **FixOut pour les données textuelles.** Nous avons étendu FixOut aux modèles entraînés sur des données textuelles. La principale originalité de cette version de FixOut réside dans la manière dont l'équité par la méconnaissance est employée, puisque la liste des mots à prendre en compte est généralement plus grande que la liste des caractéristiques sensibles dans le cadre précédent (données tabulaires).

4. **FixOut pour la classification spécifique au modèle.** Nous avons étendu le framework pour qu'il soit spécifique au modèle. En particulier, nous avons introduit une extension de FixOut pour les réseaux neuronaux (FixOut NN), en raison des récents succès très visibles des réseaux neuronaux dans plusieurs tâches.

Chapitre 1. Notions d'équité

Afin d'aborder l'équité algorithmique, diverses notions d'équité ont été définies en fonction des résultats des modèles [60, 84, 87]. Ces métriques sont décrites dans le chapitre 1 (section 1.2) et peuvent être classées en deux catégories principales. Les deux catégories stipulent que les sous-populations doivent obtenir des probabilités similaires par rapport aux résultats des modèles. Elles diffèrent par la manière dont elles divisent la population et le score qu'elles calculent pour chaque sous-population.

- Les notions d'équité de groupe (*group fairness*) divisent les sous-populations en fonction des caractéristiques sensibles uniquement, en ignorant toutes les caractéristiques non sensibles.

- Les notions d'équité individuelle (*individual fairness*) prennent également en compte les caractéristiques non sensibles, mais ils nécessitent une définition de la similarité entre deux individus pour être calculés

Une autre notion est l'équité de processus qui, au lieu de se concentrer sur les résultats finaux, est centrée sur le processus qui conduit aux résultats (classifications) [37]. L'idée principale est d'évaluer la dépendance du modèle à des caractéristiques sensibles, telles que l'origine ethnique, le genre ou l'orientation sexuelle. Cette notion particulière d'équité est présentée dans la section 1.3.

Chapitre 2. Réduction de l'iniquité

Dans le chapitre 2, nous donnons un aperçu des processeurs d'équité. Il s'agit d'approches algorithmiques (également appelées interventions algorithmiques et interventions d'amélioration

de l'équité) qui sont conçues pour optimiser une ou plusieurs notions d'équité. Ces approches sont souvent classées en fonction de l'étape à laquelle elles appliquent les interventions d'équité dans un pipeline ML.

1. *Pre-processing* (section 2.1). Ces approches (pré-processeurs) modifient l'entrée afin d'obtenir des résultats équitables. Ces processeurs peuvent être appliqués à n'importe quel modèle, puisqu'ils sont indépendants du modèle. En outre, les pré-processeurs peuvent augmenter l'incertitude du processus de classification, ce qui a un impact sur le niveau de précision [71].

2. *In-processing* (section 2.2). Ces approches (in-processeurs) tentent de modifier l'algorithme d'apprentissage pendant le processus de formation. Comme ils constituent un moyen facile d'imposer des contraintes d'équité, ces processeurs prennent généralement en compte la tension entre l'équité et les performances de classification. Cependant, ils ne peuvent pas toujours être appliqués à n'importe quel modèle, car ils sont généralement spécifiques au modèle.

3. *Post-processing* (section 2.3). Ces approches (post-processeurs) modifient les sorties de l'algorithme pour satisfaire les contraintes d'équité. Elles sont généralement agnostiques vis-à-vis du modèle et des données d'entrée, ce qui les rend plus faciles à mettre en œuvre. Cependant, les post-processeurs présentent généralement de faibles performances par rapport aux pré-processeurs [54, 71].

Dans cette thèse, nous proposons l'inclusion d'une quatrième catégorie appelée *traitement hybride* (dans la section 2.4), qui comprend les approches algorithmiques qui combinent différentes interventions d'équité en une seule méthode et, par conséquent, ne correspondent à aucune des trois catégories traditionnelles.

Chapitre 3. Méthodes d'explications

Dans le chapitre 3, nous nous sommes concentrés sur les méthodes qui génèrent des explications pour les modèles ML. Nous commençons par présenter quelques taxonomies possibles de méthodes d'explications dans la section 3.1. Ensuite, dans la section 3.2, nous nous concentrons sur des méthodes d'explication spécifiques qui produisent des explications pour des prédictions individuelles. Plus précisément, deux méthodes, LIME et SHAP, sont détaillées car elles sont nécessaires par la suite pour comprendre les contributions de la thèse, notamment sur la façon dont FixOut évalue l'équité.

LIME et SHAP partagent certaines étapes communes que les deux méthodes d'explication utilisent pour produire des explications locales pour les prédictions individuelles d'un modèle donné.

1. **Générer des instances**. À partir d'une instance de données cible x, ils génèrent plusieurs instances par perturbation. L'idée est que les instances générées constituent un nouvel ensemble qui sera ensuite utilisé pour expliquer la prédiction de l'instance cible x.

2. **Obtenir des prédictions.** Pour chaque instance générée, l'explicateur (LIME ou SHAP) utilise le modèle original pour obtenir une prédiction (par exemple, l'étiquette de classe).

3. **Construire un modèle linéaire.** Ensuite, l'explicateur ajuste un modèle linéaire pondéré g sur les prédictions obtenues. Les poids de g sont renvoyés comme importance de la caractéristique.

Malgré les points communs mentionnés ci-dessus, LIME et SHAP diffèrent dans la définition du *kernel* qui définit le voisinage de x et également dans la fonction de complexité (*regularizer*) utilisée pour produire les explications.

Ensuite, dans la section 3.3, nous présentons quelques approches permettant d'obtenir des explications globales à partir d'explications locales. LIME et SHAP ne fournissent des explications que pour les prédictions individuelles. Cependant, afin de détecter les biais inattendus, une compréhension globale du fonctionnement interne du modèle ML peut être nécessaire. Nous nous concentrons donc sur une stratégie permettant d'obtenir des explications globales à partir d'explications individuelles, appelée SP [72]. Elle consiste à sélectionner un sous-ensemble d'explications locales qui aide à interpréter le comportement global d'un modèle ML donné. L'idée principale sur laquelle s'appuie SP est d'échantillonner un ensemble d'instances dont les explications ne sont pas redondantes et qui a une "couverture élevée".

Dans la section 3.4, nous soulignons certains inconvénients de l'utilisation de LIME et SHAP. Ensuite, dans la section 3.5, nous mentionnons de nouvelles extensions de LIME et SHAP pour surmonter les problèmes mentionnés précédemment. Enfin, nous terminons le chapitre dans la section 3.6 en présentant une méthode d'explication qui, contrairement à LIME et SHAP, est spécifique au modèle. Cet explicateur s'appelle PathExplain et se concentre sur l'explication des modèles différentiables, tels que les modèles d'apprentissage profond. Cette méthode d'explication est ensuite utilisée dans une version de FixOut spécifique au modèle.

Chapitre 4. Réduction de l'iniquité par méconnaissance

Dans le chapitre 4, nous présentons FixOut, un framework qui s'attaque aux deux principaux problèmes de l'équité algorithmique, c'est-à-dire (1) l'évaluation de l'équité et (2) la réduction de l'iniquité. FixOut utilise des explications pour évaluer l'équité du processus dans la phase d'évaluation de l'équité. Afin d'atténuer l'iniquité, le framework pousse plus loin l'équité par l'ignorance en combinant deux interventions d'équité différentes sans compromettre les performances de classification. FixOut supprime les caractéristiques sensibles avant l'entraînement des classificateurs et modifie l'entrée qui caractérise la phase de prétraitement. Le framework produit un pool de classificateurs dont les sorties sont combinées grâce à une fonction d'agrégation. Cette fonction manipule les frontières de décision des classifieurs afin d'appliquer l'équité, ce qui caractérise la phase de post-traitement.

Afin de détailler le framework, nous commençons le chapitre, dans la section 4.1, en reliant l'équité des processus et l'explicabilité. Nous présentons ensuite un aperçu des composants de FixOut dans la section 4.2. Dans la sous-section 4.2.1, nous détaillons comment FixOut évalue l'équité en employant des explications locales post-hoc dans la première composante du framework, puisque FixOut va plus loin que les méthodes existantes en permettant l'utilisation de toute méthode d'explication qui produit des explications sous la forme d'importance des caractéristiques.

Nous présentons ensuite FixOut comme un processeur d'équité hybride dans la sous-section 4.2.2, c'est-à-dire le deuxième composant du framework qui est responsable de l'atténuation de l'iniquité en combinant des interventions d'équité avant et après le traitement. Plus précisément, dans l'intervention de post-traitement, nous incorporons différentes fonctions d'agrégation dans FixOut. Par exemple, l'une de ces fonctions prend en compte la contribution des caractéristiques sensibles. Elle effectue une agrégation pondérée en attribuant des poids élevés aux classificateurs qui ont été formés sans les caractéristiques sensibles à forte contribution.

Nous concluons ce chapitre en présentant un exemple complet d'application de FixOut dans

un ensemble de données du monde réel dans la section 4.4 et en discutant des avantages, des limites et des perspectives du framework générique dans la section 4.5.

Chapitre 5. Réduction de l'iniquité en cas de données tabulaires

Dans le chapitre 5, nous instancions FixOut pour aborder le problème de la réduction de l'iniquité sur des modèles formés sur des données tabulaires. Notre objectif principal est de vérifier si FixOut est capable de réduire l'iniquité, du point de vue de l'équité des processus, sans compromettre les performances de classification. Comme FixOut est un framework générique qui demande aux praticiens de ML de faire des choix tels que la méthode d'explication, le nombre maximum de caractéristiques à considérer comme importantes et le type de fonction d'agrégation, nous définissons quatre instances de FixOut: deux d'entre elles utilisent LIME et les deux autres SHAP. Ces quatre instances de FixOut sont utilisées dans les expériences afin de répondre aux questions suivantes

Q1 Y a-t-il des différences entre les FixOut avec des explications LIME et les FixOut avec des explications SHAP ?
Nous abordons les problèmes d'iniquité en montrant que LimeOut peut être adapté à différentes méthodes d'explication et fonctions d'agrégation. FixOut étend LimeOut en autorisant toute méthode d'explication basée sur l'importance des caractéristiques. Pour illustrer, nous considérons FixOut instancié par SHAP et LIME pour évaluer l'équité du processus.

Q2 Quel est l'impact de la fonction d'agrégation sur la performance des modèles de sortie de FixOut?
Pour construire le modèle d'ensemble final, FixOut peut utiliser différentes fonctions d'agrégation (par exemple, la moyenne simple ou la moyenne pondérée). Nous instancions FixOut avec l'une des fonctions d'agrégation présentées dans la section 4.2.2.

Q3 FixOut améliore-t-elle les mesures d'équité standard ?
Nous avons conçu FixOut pour améliorer l'équité des processus, qui est une notion d'équité particulière. Cependant, la littérature rapporte diverses définitions de l'équité, nous évaluons donc FixOut sur les notions d'équité standard bien connues afin de vérifier empiriquement si nous observons également des impacts bénéfiques sur ces métriques. L'étude empirique sur les notions d'équité offre une perspective intéressante de FixOut, car elle permet de découvrir l'iniquité des modèles à travers le prisme de notions d'équité bien connues.

Q4 Comment réduire l'intervention humaine dans l'utilisation de FixOut sur des données tabulaires ?
FixOut utilise des méthodes d'explication agnostiques au modèle pour évaluer l'équité du processus. Il requiert au préalable que les utilisateurs définissent la taille d'échantillonnage des instances de données et le nombre de caractéristiques utilisées pour évaluer l'équité des modèles pré-entraînés, ce qui soulève le besoin d'automatisation ou, au moins, de réglage fin. Nous proposons ensuite un algorithme qui sélectionne automatiquement le nombre de caractéristiques utilisées par FixOut pour évaluer l'équité.

Nous commençons le chapitre 5 en décrivant les ensembles de données tabulaires que nous avons utilisés pendant les expériences dans la section 5.1. Nous expliquons ensuite les étapes de

prétraitement et le dispositif expérimental que nous avons appliqué dans la section 5.2. Ensuite, dans la section 5.3, nous analysons les résultats expérimentaux sous différents angles, à savoir : l'évaluation de l'équité du processus, les performances de classification et l'évaluation des mesures d'équité. Dans la section 5.4, nous proposons une méthode pour automatiser le choix du paramètre k requis par FixOut et nous présentons également les résultats expérimentaux obtenus lorsque la méthode proposée est utilisée avec FixOut. Enfin, nous terminons ce chapitre en discutant des travaux futurs pour ces instanciations de FixOut (section 5.5).

Chapitre 6. Réduction de l'iniquité en cas de données textuelles

Dans le chapitre 6, nous étendons FixOut pour qu'il puisse être appliqué à des modèles de données textuelles et nous appliquons cette extension de FixOut à la tâche de classification des tweets comme discours haineux ou non [5]. L'idée est de démontrer l'adaptabilité du framework FixOut à un autre type de données, en l'occurrence des données textuelles. L'hypothèse est que FixOut *est capable d'améliorer l'équité du processus non seulement dans les données tabulaires, mais aussi dans les données textuelles.* Nous l'abordons donc en étudiant la question de recherche suivante

Q5 FixOut réduit-il la dépendance des modèles aux mots sensibles des modèles formés sur des données textuelles ?
Nous tirons parti de la proposition FixOut qui traite de l'iniquité algorithmique par le biais de la méconnaissance, et l'étendons à d'autres contextes, à savoir des scénarios de classification traitant des données textuelles. Pour cela, nous commençons par adapter la notion d'abandon de caractéristiques à l'abandon de mots, puis à la stratégie d'ensemble. Étant donné que nous avons plusieurs mots à considérer comme sensibles, nous l'étendons ensuite à une approche d'abandon par sac de mots afin de comparer l'impact de la suppression de plusieurs mots en même temps. Essentiellement, l'idée de l'abandon d'un sac de mots est de supprimer un ensemble de mots plutôt qu'un seul, de sorte que nous avons moins de classificateurs à entraîner et aussi moins de classificateurs qui prennent en compte un mot sensible.

Bibliography

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9525–9536, 2018.

[2] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna M. Wallach. A reductions approach to fair classification. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 60–69. PMLR, 2018.

[3] Alekh Agarwal, Miroslav Dudík, and Zhiwei Steven Wu. Fair regression: Quantitative definitions and reduction-based algorithms. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 120–129. PMLR, 2019.

[4] Chirag Agarwal, Eshika Saxena, Satyapriya Krishna, Martin Pawelczyk, Nari Johnson, Isha Puri, Marinka Zitnik, and Himabindu Lakkaraju. Openxai: Towards a transparent evaluation of model explanations. *CoRR*, abs/2206.11104, 2022.

[5] Guilherme Alves, Maxime Amblard, Fabien Bernier, Miguel Couceiro, and Amedeo Napoli. Reducing unintended bias of ML models on tabular and textual data. In *8th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2021, Porto, Portugal, October 6-9, 2021*, pages 1–10. IEEE, 2021.

[6] Guilherme Alves, Fabien Bernier, Miguel Couceiro, and Amedeo Napoli. FixOut: an ensemble approach to fairer models. *HAL preprint hal:03033181*, 2020.

[7] Julia Angwin et al. Machine bias: There's software used across the country to predict future criminals. *And it's biased against blacks. ProPublica*, 23, 2016.

[8] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. http://www.fairmlbook.org.

[9] Yahav Bechavod and Katrina Ligett. Learning fair classifiers: A regularization-inspired approach. *CoRR*, abs/1707.00044, 2017.

[10] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John T. Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI

fairness 360: An extensible toolkit for detecting, understanding, and mitigating unwanted algorithmic bias. *CoRR*, abs/1810.01943, 2018.

[11] João Bento, Pedro Saleiro, André Ferreira Cruz, Mário A. T. Figueiredo, and Pedro Bizarro. Timeshap: Explaining recurrent models through sequence perturbations. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 2565–2573. ACM, 2021.

[12] Vaishnavi Bhargava, Miguel Couceiro, and Amedeo Napoli. Limeout: An ensemble approach to improve process fairness. In *ECML PKDD 2020 Workshops - Workshops of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, September 14-18, 2020, Proceedings*, volume 1323 of *Communications in Computer and Information Science*, pages 475–491. Springer, 2020.

[13] Steven Bramhall, Hayley Horn, Michael Tieu, and Nibhrat Lohia. Qlime-a quadratic local interpretable model-agnostic explanation approach. *SMU Data Science Review*, 3(1):4, 2020.

[14] Pavel Brazdil, Christophe G. Giraud-Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning - Applications to Data Mining*. Cognitive Technologies. Springer, 2009.

[15] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[16] Leo Breiman. Statistical modeling: The two cultures. *Statistical science*, 16(3):199–231, 2001.

[17] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *ICDM Workshops 2009, IEEE International Conference on Data Mining Workshops, Miami, Florida, USA, 6 December 2009*, pages 13–18. IEEE Computer Society, 2009.

[18] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Min. Knowl. Discov.*, 21(2):277–292, 2010.

[19] Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. Generating hierarchical explanations on text classification via feature interaction detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 5578–5593. Association for Computational Linguistics, 2020.

[20] A. Feder Cooper, Ellen Abrams, and Na Na. Emergent unfairness in algorithmic fairness-accuracy trade-off research. In *AIES '21: AAAI/ACM Conference on AI, Ethics, and Society, Virtual Event, USA, May 19-21, 2021*, pages 46–54. ACM, 2021.

[21] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *CoRR*, abs/1808.00023, 2018.

[22] Amanda Coston, Ashesh Rambachan, and Alexandra Chouldechova. Characterizing fairness over the set of good models under selective labels. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2144–2155. PMLR, 2021.

[23] Thomas Davidson, Debasmita Bhattacharya, and Ingmar Weber. Racial bias in hate speech and abusive language detection datasets. *CoRR*, abs/1905.12516, 2019.

[24] Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017*, pages 512–515. AAAI Press, 2017.

[25] Julien Delaunay, Luis Galárraga, and Christine Largouët. Improving anchor-based explanations. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 3269–3272. ACM, 2020.

[26] Botty Dimanov, Umang Bhatt, Mateja Jamnik, and Adrian Weller. You shouldn't trust me: Learning models which conceal unfairness from multiple explanation methods. In *ECAI 2020 - 24th European Conference on Artificial Intelligence, 29 August-8 September 2020, Santiago de Compostela, Spain, August 29 - September 8, 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2473–2480. IOS Press, 2020.

[27] Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*, pages 67–73. ACM, 2018.

[28] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard S. Zemel. Fairness through awareness. In *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 214–226. ACM, 2012.

[29] Benjamin Fish, Jeremy Kun, and Ádám Dániel Lelkes. A confidence-based approach for balancing fairness and accuracy. In *Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, Florida, USA, May 5-7, 2016*, pages 144–152. SIAM, 2016.

[30] Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20:177:1–177:81, 2019.

[31] Alex Alves Freitas. Comprehensible classification models: a position paper. *SIGKDD Explor.*, 15(1):1–10, 2013.

[32] Sorelle A. Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. On the (im)possibility of fairness. *CoRR*, abs/1609.07236, 2016.

[33] Sorelle A. Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P. Hamilton, and Derek Roth. A comparative study of fairness-enhancing interventions in machine learning. *CoRR*, abs/1802.04422, 2018.

[34] Damien Garreau and Ulrike von Luxburg. Explaining the explainer: A first theoretical analysis of LIME. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 1287–1296. PMLR, 2020.

[35] Romaric Gaudel, Luis Galárraga, Julien Delaunay, Laurence Rozé, and Vaishnavi Bhargava. s-lime: Reconciling locality and fidelity in linear explanations. In *Advances in Intelligent Data Analysis XX - 20th International Symposium on Intelligent Data Analysis, IDA 2022, Rennes, France, April 20-22, 2022, Proceedings*, volume 13205 of *Lecture Notes in Computer Science*, pages 102–114. Springer, 2022.

[36] Amirata Ghorbani, James Wexler, James Y. Zou, and Been Kim. Towards automatic concept-based explanations. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9273–9282, 2019.

[37] Nina Grgic-Hlaca, Muhammad Bilal Zafar, Krishna P Gummadi, and Adrian Weller. The case for process fairness in learning: Feature selection for fair decision making. In *NIPS Symposium on Machine Learning and the Law*, page 2, 2016.

[38] Dominique Guegan et al. Credit risk analysis using machine and deep learning models. volume 6, page 38 pages, 2018.

[39] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

[40] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. Local rule-based explanations of black box decision systems. *CoRR*, abs/1805.10820, 2018.

[41] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51(5):93:1–93:42, 2019.

[42] Tessa Han, Suraj Srinivas, and Himabindu Lakkaraju. Which explanation should I choose? A function approximation perspective to characterizing post hoc explanations. *CoRR*, abs/2206.01254, 2022.

[43] Dominik Hangartner, Daniel Kopp, and Michael Siegenthaler. Monitoring hiring discrimination through online recruitment platforms. *Nature*, 589(7843):572–576, 2021.

[44] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3315–3323, 2016.

[45] Vasileios Iosifidis, Besnik Fetahu, and Eirini Ntoutsi. FAE: A fairness-aware ensemble framework. In *2019 IEEE International Conference on Big Data (IEEE BigData), Los Angeles, CA, USA, December 9-12, 2019*, pages 1375–1380. IEEE, 2019.

[46] Vasileios Iosifidis and Eirini Ntoutsi. Adafair: Cumulative fairness adaptive boosting. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, pages 781–790. ACM, 2019.

[47] Joseph D. Janizek, Pascal Sturmfels, and Su-In Lee. Explaining explanations: Axiomatic feature interactions for deep networks. *Journal of Machine Learning Research*, 22:104:1–104:54, 2021.

[48] Sérgio M. Jesus, Catarina Belém, Vladimir Balayan, João Bento, Pedro Saleiro, Pedro Bizarro, and João Gama. How can I choose an explainer?: An application-grounded evaluation of post-hoc explanations. In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*, pages 805–815. ACM, 2021.

[49] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *12th IEEE International Conference on Data Mining, ICDM 2012, Brussels, Belgium, December 10-13, 2012*, pages 924–929. IEEE Computer Society, 2012.

[50] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II*, volume 7524 of *Lecture Notes in Computer Science*, pages 35–50. Springer, 2012.

[51] Been Kim, Oluwasanmi Koyejo, and Rajiv Khanna. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2280–2288, 2016.

[52] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 2673–2682. PMLR, 2018.

[53] Jon M. Kleinberg, Sendhil Mullainathan, and Manish Raghavan. Inherent trade-offs in the fair determination of risk scores. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, volume 67 of *LIPIcs*, pages 43:1–43:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[54] Nikita Kozodoi, Johannes Jacob, and Stefan Lessmann. Fairness in credit scoring: Assessment, implementation and profit implications. *Eur. J. Oper. Res.*, 297(3):1083–1094, 2022.

[55] Michelle Seng Ah Lee and Luciano Floridi. Algorithmic fairness in mortgage lending: from absolute conditions to relational trade-offs. *Minds Mach.*, 31(1):165–191, 2021.

[56] Michelle Seng Ah Lee, Luciano Floridi, and Jatinder Singh. Formalising trade-offs beyond algorithmic fairness: lessons from ethical philosophy and welfare economics. *AI and Ethics*, 1(4):529–544, 2021.

[57] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220, 2016.

[58] Zachary C. Lipton, Julian J. McAuley, and Alexandra Chouldechova. Does mitigating ml's impact disparity require treatment disparity? In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8136–8146, 2018.

[59] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4765–4774, 2017.

[60] Karima Makhlouf, Sami Zhioua, and Catuscia Palamidessi. On the applicability of ML fairness notions. *CoRR*, abs/2006.16745, 2020.

[61] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6):115:1–115:35, 2021.

[62] Aditya Krishna Menon and Robert C. Williamson. The cost of fairness in binary classification. In *Conference on Fairness, Accountability and Transparency, FAT 2018, 23-24 February 2018, New York, NY, USA*, volume 81 of *Proceedings of Machine Learning Research*, pages 107–118. PMLR, 2018.

[63] Shira Mitchell, Eric Potash, Solon Barocas, Alexander D'Amour, and Kristian Lum. Prediction-based decisions and fairness: A catalogue of choices, assumptions, and definitions. *arXiv preprint arXiv:1811.07867*, 2018.

[64] Shira Mitchell, Eric Potash, Solon Barocas, Alexander D'Amour, and Kristian Lum. Algorithmic fairness: Choices, assumptions, and definitions. *Annual Review of Statistics and Its Application*, 8:141–163, 2021.

[65] C. Molnar. Interpretable machine learning: A guide for making black box models explainable. https://christophm.github.io/interpretable-ml-book/, 2018. [Online; accessed 26-November-2020].

[66] Johannes J A Moors. The meaning of kurtosis: Darlington reexamined. *The American Statistician*, 40(4):283–284, 1986.

[67] Executive Office of the President, Cecilia Munoz, Domestic Policy Council Director, Megan (US Chief Technology Officer Smith (Office of Science, Technology Policy)), DJ (Deputy Chief Technology Officer for Data Policy, Chief Data Scientist Patil (Office of Science, and Technology Policy)). *Big data: A report on algorithmic systems, opportunity, and civil rights*. Executive Office of the President, 2016.

[68] Kalia Orphanou, Jahna Otterbacher, Styliani Kleanthous, Khuyagbaatar Batsuren, Fausto Giunchiglia, Veronika Bogina, Avital Shulner Tal, Alan Hartman, and Tsvi Kuflik. Mitigating bias in algorithmic systems: A fish-eye view of problems and solutions across domains. *CoRR*, abs/2103.16953, 2021.

[69] Orestis Papakyriakopoulos, Simon Hegelich, Juan Carlos Medina Serrano, and Fabienne Marco. Bias in word embeddings. In *FAT* '20: Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, January 27-30, 2020*, pages 446–457. ACM, 2020.

[70] Dana Pessach and Erez Shmueli. Improving fairness of artificial intelligence algorithms in privileged-group selection bias data settings. *Expert Syst. Appl.*, 185:115667, 2021.

[71] Dana Pessach and Erez Shmueli. A review on fairness in machine learning. *ACM Computing Surveys (CSUR)*, 55(3):1–44, 2022.

[72] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. ACM, 2016.

[73] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1527–1535. AAAI Press, 2018.

[74] Yuji Roh, Kangwook Lee, Steven Euijong Whang, and Changho Suh. Fairbatch: Batch selection for model fairness. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[75] Ricardo Salazar, Felix Neutatz, and Ziawasch Abedjan. Automated feature engineering for algorithmic fairness. *Proc. VLDB Endow.*, 14(9):1694–1702, 2021.

[76] Babak Salimi, Luke Rodriguez, Bill Howe, and Dan Suciu. Interventional fairness: Causal database repair for algorithmic fairness. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, pages 793–810. ACM, 2019.

[77] Gilbert Saporta. *Probabilités, analyse des données et statistique*. Editions Technip, 2006.

[78] Lloyd S Shapley. A value for n-person games. In *Contributions to the Theory of Games*, pages 307–317, 1953.

[79] Radwa El Shawi, Youssef Sherif, Mouaz H. Al-Mallah, and Sherif Sakr. ILIME: local and global interpretable model-agnostic explainer of black-box decision. In *Advances in Databases and Information Systems - 23rd European Conference, ADBIS 2019, Bled, Slovenia, September 8-11, 2019, Proceedings*, volume 11695 of *Lecture Notes in Computer Science*, pages 53–68. Springer, 2019.

[80] Dylan Slack, Anna Hilgard, Sameer Singh, and Himabindu Lakkaraju. Reliable post hoc explanations: Modeling uncertainty in explainability. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 9391–9404, 2021.

[81] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: adversarial attacks on post hoc explanation methods. In *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, pages 180–186. ACM, 2020.

[82] Dylan Slack, Sophie Hilgard, Sameer Singh, and Himabindu Lakkaraju. Feature attributions and counterfactual explanations can be manipulated. *CoRR*, abs/2106.12563, 2021.

[83] Kacper Sokol, Alexander Hepburn, Raúl Santos-Rodríguez, and Peter A. Flach. blimey: Surrogate prediction explanations beyond LIME. *CoRR*, abs/1910.13016, 2019.

[84] Till Speicher, Hoda Heidari, Nina Grgic-Hlaca, Krishna P. Gummadi, Adish Singla, Adrian Weller, and Muhammad Bilal Zafar. A unified approach to quantifying algorithmic unfairness: Measuring individual &group unfairness via inequality indices. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 2239–2248. ACM, 2018.

[85] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

[86] Michael L. Wick, Swetasudha Panda, and Jean-Baptiste Tristan. Unlocking fairness: a trade-off revisited. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8780–8789, 2019.

[87] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P. Gummadi. Fairness constraints: Mechanisms for fair classification. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 962–970. PMLR, 2017.

[88] Muhammad Rehman Zafar and Naimul Mefraz Khan. DLIME: A deterministic local interpretable model-agnostic explanations approach for computer-aided diagnosis systems. *CoRR*, abs/1906.10263, 2019.

[89] Mohammed J. Zaki and Wagner Meira Jr. *Data Mining and Analysis: Fundamental Concepts and Algorithms*. Cambridge University Press, 2014.

[90] Richard S. Zemel, Yu Wu, Kevin Swersky, Toniann Pitassi, and Cynthia Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 325–333. JMLR.org, 2013.

[91] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2018, New Orleans, LA, USA, February 02-03, 2018*, pages 335–340. ACM, 2018.

[92] Indre Zliobaite. On the relation between accuracy and fairness in binary classification. *CoRR*, abs/1505.05723, 2015.

Résumé

Les décisions algorithmiques sont actuellement utilisées quotidiennement. Ces décisions reposent souvent sur des algorithmes d'apprentissage automatique (*machine learning* – ML) qui peuvent produire des modèles complexes et opaques. Des études récentes ont soulevé des problèmes d'iniquité en révélant des résultats discriminatoires produits par les modèles ML contre des minorités et des groupes non privilégiés. Comme les modèles ML sont capables d'amplifier la discrimination en raison de résultats injustes, cela révèle la nécessité d'approches qui découvrent et suppriment les biais inattendues.

L'évaluation de l'équité et l'atténuation de l'iniquité sont les deux tâches principales qui ont motivé la croissance du domaine de recherche en *équité algorithmique* (*algorithmic fairness*). Plusieurs notions utilisées pour évaluer l'équité se concentrent sur les résultats et sont liées à des caractéristiques sensibles (par exemple, le sexe et l'éthinicité) par des mesures statistiques. Bien que ces notions aient une sémantique distincte, l'utilisation de ces définitions de l'équité est critiquée pour sa compréhension réductrice de l'équité, dont le but est essentiellement de mettre en œuvre des rapports d'acceptation/non-acceptation, ignorant d'autres perspectives sur l'inégalité et l'impact sociétal. *Process fairness* (équité des procédures) est au contraire une notion d'équité subjective, centrée sur le processus qui conduit aux résultats.

Pour atténuer ou supprimer l'iniquité, les approches appliquent généralement des interventions en matière d'équité selon des étapes spécifiques. Elles modifient généralement soit (1) les données avant l'apprentissage, soit (2) la fonction d'optimisation, soit (3) les sorties des algorithmes afin d'obtenir des résultats plus équitables. Récemment, les recherches sur l'équité algorithmique ont été consacrées à l'exploration de combinaisons de différentes interventions en matière d'équité, ce qui est désigné dans cette thèse par le *traitement hybride de l'équité*. Une fois que nous essayons d'atténuer l'iniquité, une tension entre l'équité et la performance apparaît, connue comme le compromis équité/précision.

Cette thèse se concentre sur le problème du compromis équité/précision, puisque nous sommes intéressés par la réduction des biais inattendues sans compromettre les performances de classification. Nous proposons donc des méthodes basées sur les ensembles pour trouver un bon compromis entre l'équité et la performance de classification des modèles ML, en particulier les modèles de classification binaire. De plus, ces méthodes produisent des classifieurs d'ensemble grâce à une combinaison d'interventions sur l'équité, ce qui caractérise les approches de traitement hybride de l'équité.

Nous proposons FixOut (*FaIrness through eXplanations and feature dropOut*), un framework centré sur l'humain et agnostique vis-à-vis des modèles qui améliore l'équité des processus sans compromettre les performances de classification. Il reçoit en entrée un classificateur pré-entraîné (modèle original), un ensemble de données, un ensemble de caractéristiques sensibles et une méthode d'explication, et il produit un nouveau classificateur qui dépend moins des caractéristiques sensibles. Pour évaluer la dépendance d'un modèle pré-entraîné aux caractéristiques sensibles, FixOut utilise des explications pour estimer la contribution des caractéristiques aux résultats des modèles. S'il s'avère que les caractéristiques sensibles contribuent globalement aux résultats des modèles, alors le modèle est considéré comme injuste. Dans ce cas, il construit un groupe de classificateurs plus justes qui sont ensuite agrégés pour obtenir un classificateur d'ensemble. Nous montrons l'adaptabilité de FixOut sur différentes combinaisons de méthodes

d'explication et d'approches d'échantillonnage. Nous évaluons également l'efficacité de FixOut par rapport au *process fairness* mais aussi en utilisant des notions d'équité standard bien connues disponibles dans la littérature. De plus, nous proposons plusieurs améliorations telles que l'automatisation du choix des paramètres de FixOut et l'extension de FixOut à d'autres types de données.

Mots-clés: Évaluation de l'équité, atténuation de l'iniquité, explications, compromis équité/exactitude.

Abstract

Algorithmic decisions are currently being used on a daily basis. These decisions often rely on Machine Learning (ML) algorithms that may produce complex and opaque ML models. Recent studies raised unfairness concerns by revealing discriminating outcomes produced by ML models against minorities and unprivileged groups. As ML models are capable of amplifying discrimination against minorities due to unfair outcomes, it reveals the need for approaches that uncover and remove unintended biases.

Assessing fairness and mitigating unfairness are the two main tasks that have motivated the growth of the research field called *algorithmic fairness*. Several notions used to assess fairness focus on the outcomes and link to sensitive features (e.g. gender and ethnicity) through statistical measures. Although these notions have distinct semantics, the use of these definitions of fairness is criticized for being a reductionist understanding of fairness whose aim is basically to implement accept/not-accept reports, ignoring other perspectives on inequality and on societal impact. Process fairness instead is a subjective fairness notion which is centered on the process that leads to outcomes. To mitigate or remove unfairness, approaches generally apply fairness interventions in specific steps. They usually change either (1) the data before training or (2) the optimization function or (3) the algorithms' outputs in order to enforce fairer outcomes. Recently, research on algorithmic fairness have been dedicated to explore combinations of different fairness interventions, which is referred to in this thesis as *fairness hybrid-processing*. Once we try to mitigate unfairness, a tension between fairness and performance arises that is known as the fairness-accuracy trade-off.

This thesis focuses on the fairness-accuracy trade-off problem since we are interested in reducing unintended biases without compromising classification performance. We thus propose ensemble-based methods to find a good compromise between fairness and classification performance of ML models, in particular models for binary classification. In addition, these methods produce ensemble classifiers thanks to a combination of fairness interventions, which characterizes the fairness hybrid-processing approaches.

We introduce FixOut (FaIrness through eXplanations and feature dropOut), the human centered, model-agnostic framework that improves process fairness without compromising classification performance. It receives a pre-trained classifier (original model), a dataset, a set of sensitive features, and an explanation method as input, and it outputs a new classifier that is less reliant on the sensitive features. To assess the reliance of a given pre-trained model on sensitive features, FixOut uses explanations to estimate the contribution of features to models' outcomes. If sensitive features are shown to contribute globally to models' outcomes, then the model is deemed unfair. In this case, it builds a pool of fairer classifiers that are then aggregated to obtain an ensemble classifier. We show the adaptability of FixOut on different combinations of explanation methods and sampling approaches. We also evaluate the effectiveness of FixOut w.r.t. to process fairness but also using well-known standard fairness notions available in the literature. Furthermore, we propose several improvements such as automating the choice of FixOut 's parameters and extending FixOut to other data types.

Keywords: Fairness assessment, unfairness mitigation, explanations, fairness-accuracy trade-off.