

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Responsible AI - How data-based decision making can increase bias in smart manufacturing

João Alberto Preto Rodrigues Praça



Mestrado em Engenharia Informática e Computação

Supervisor: João Pedro Correia dos Reis

Second Supervisor: Gil Manuel Magalhães de Andrade Gonçalves

October 31, 2022

Responsible AI - How data-based decision making can increase bias in smart manufacturing

João Alberto Preto Rodrigues Praça

Mestrado em Engenharia Informática e Computação

Approved by:

President: Luis F. Teixeira

Referee: Fátima Leal

Referee: João Reis

October 31, 2022

Abstract

Given the exponential growth of Artificial Intelligence over the recent years, it has become increasingly important to understand the decisions of an intelligent system in a transparent manner, namely in the law and insurance areas, where the need to treat everyone equally regardless of their gender, race, sexual orientation, etc., is of utmost importance. Explainability allows determining the variables that led to a certain decision and is crucial to detect and tackling issues such as bias. This need to understand the decisions behind a Machine Learning model can also be applied in manufacturing, in cases of defect detection to trace back the origin of the defect, so that it can be prevented going forward.

Bias detection is a major focus and concern in Explainable AI algorithms nowadays since the adoption of AI algorithms has been very broad, so any small favoring or prejudice towards a group becomes a massive problem. This concept has increasingly been explored in areas that involve people, such as finances or healthcare, where no one should be discriminated by characteristics that don't make a difference in the final decision, including gender and race. In manufacturing, however, there is barely any information surrounding the topic. In a sector full of different types of equipment, brands, and processes, detecting faulty units, uncovering the reason behind those flaws, as well as any tendency towards a specific anomaly or defect, and how to prevent them could be a great advantage and an opportunity to decrease waste, time and expenses.

This work will focus on studying the effects of bias in data-based machine learning models, in situations where it is not possible to correct it while maintaining the model's accuracy, as it's not clear if the tendency is in fact bias or just an attribute of the data. Data-based models are used to make decisions, and these decisions will generate a new dataset, that will then be used to retrain the model. If bias is present in the first dataset used, every time the model is retrained the bias can be amplified, to a point where it won't be detectable. In this case, the portion of the dataset that causes the bias will grow and eventually become the majority of the dataset, thus turning the bias into the norm. This effect was analyzed using two biased manufacturing process datasets and was named the "magnification effect", similar to the error propagation problem in statistics. The work developed comprised two approaches to simulate the retraining process of a prediction system in a production line is subjected to, as well as a comparison between both.

The results showed that bias can in fact increase exponentially when a biased dataset is retrained using the predictions it generates, and it also affects the performance of the model negatively. However, this effect wasn't observed in all experiments, being highly dependent on the dataset and machine learning model used.

Keywords: Smart Manufacturing, Explainable Artificial Intelligence, Bias, Magnification Effect

Resumo

Dado o crescimento exponencial da Inteligência Artificial ao longo dos últimos anos, torna-se cada vez mais importante compreender as decisões de um sistema inteligente de forma transparente, nomeadamente nas áreas do direito e seguros, onde a necessidade de tratar todos de forma igual independentemente do seu género, raça, orientação sexual, etc., é de extrema importância. A explicabilidade permite determinar as variáveis que levaram a uma determinada decisão e é crucial para detectar e abordar questões como o *bias*. Esta necessidade de entender as decisões por trás de um modelo de Machine Learning também pode ser aplicada na área industrial, em casos de detecção de defeitos para investigar a origem do defeito, para que possa ser evitado no futuro.

A detecção de *bias* é um grande foco e preocupação nos algoritmos de explicabilidade em Inteligência Artificial (IA) hoje em dia, já que a adoção de algoritmos de IA tem sido muito ampla, de modo que qualquer pequeno favorecimento ou preconceito em relação a um grupo se torna um grande problema. Este conceito tem sido cada vez mais explorado em áreas que envolvem humanos, como a área financeira ou da saúde, onde ninguém deve ser discriminado por características que não fazem efetivamente diferença na decisão final, incluindo género e raça. Na indústria, no entanto, quase não há informações sobre este assunto. Num setor repleto de diferentes tipos de máquinas, marcas e processos, detectar produtos defeituosos, descobrir o motivo dessas falhas, bem como qualquer tendência a uma anomalia ou defeito específico, e como evitá-las, pode ser uma grande vantagem e uma oportunidade para diminuir o desperdício, tempo e despesas.

Este trabalho centrou-se em estudar os efeitos do *bias* em modelos de *Machine Learning* baseados em dados, em situações em que não é possível corrigi-lo mantendo a precisão do modelo, pois não está claro se a tendência é de fato *bias* ou apenas uma característica dos dados. Modelos baseados em dados são usados para tomar decisões, e essas decisões irão gerar um novo conjunto de dados, que será usado para treinar novamente o modelo. Se o *bias* estiver presente no primeiro conjunto de dados usado para treino, cada vez que o modelo for treinado novamente o *bias* poderá ser amplificado, até um ponto em que não será detetável. Nesse caso, a porção do conjunto de dados que causa o *bias* crescerá e eventualmente tornar-se-á na maioria do conjunto de dados, tornando o *bias* a norma. Este efeito foi analisado usando dois conjuntos de dados de processos de fabricação tendenciosos e foi denominado “efeito de magnificação”, semelhante ao problema de propagação do erro em estatística. O trabalho desenvolvido compreendeu duas abordagens para simular o processo de retreinamento de um sistema de previsão numa linha de produção, bem como uma comparação entre ambas.

Os resultados mostraram que o *bias* pode, de facto, aumentar exponencialmente quando um conjunto de dados tendencioso é treinado novamente usando as previsões que ele gera e também que este afeta negativamente o desempenho do modelo. No entanto, esse efeito não foi observado em todos as experiências efetuadas, sendo altamente dependente do conjunto de dados e do modelo

de *Machine Learning* utilizado.

Keywords: Indústria Inteligente, Explicabilidade em Inteligência Artificial, Bias, Efeito de Magnificação

Acknowledgements

I would like to thank my supervisors, Professor João Reis and Professor Gil Gonçalves, and also Artur Freitas, from the DIGI2 lab, for all the support, guidance, patience and availability throughout these past months while working on this dissertation. They have provided me with resources, constant advice and key knowledge that made this process more straightforward.

I would also like to thank my family and friends for always supporting me and believing in me even when I don't. Thank you for being my safe haven and for making life a little less overwhelming.

João Praça

“You are always a student, never a master.”

Conrad Hall

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	1
1.3	Objectives and Research Questions	2
1.4	Document Structure	2
2	Literature Review	3
2.1	Explainable AI	3
2.1.1	White-box and Black-box models	3
2.1.2	Methods	4
2.2	Bias	5
2.2.1	Detecting Bias	5
2.2.2	Minimizing Bias	7
3	Methodology and Implementation	10
3.1	Problem Definition	10
3.2	Datasets	10
3.2.1	Plastic Film Defects	11
3.2.2	Fabric White Dyeing	11
3.3	Retraining simulation Approaches	13
3.3.1	First Approach	13
3.3.2	Second Approach	15
4	Experiments and Results	16
4.1	Plastic Film Defects	16
4.1.1	Experiments	16
4.1.2	Results	16
4.2	Fabric Dyeing	20
4.2.1	Experiments	20
4.2.2	Results	21
4.3	Summary	23
5	Conclusions and Future Work	26
5.1	Conclusions	26
5.2	Future Work	27
	References	28

List of Figures

2.1	Example of complex model and LIME's surrogate model [5]	4
2.2	Example of SHAP subsets, in the context of a prize division within a group of people [6]	5
3.1	Correlation matrix for the Fabric White Dyeing dataset	12
3.2	First Approach diagram	14
3.3	Second Approach diagram	15
4.1	First approach - SHAP values of Cycle 1/50 using Random Forest	17
4.2	First approach - SHAP values of Cycle 10/50 using Random Forest	17
4.3	First approach - Accuracy and F1 score of the 10 Cycles test using Random Forest	18
4.4	First approach - Accuracy and F1 score of the 50 Cycles test using Random Forest	18
4.5	First approach - Accuracy and F1 score of the 100 Cycles test using Random Forest	19
4.6	First approach - SHAP values of Cycle 1/10 using XGBoost	19
4.7	First approach - SHAP values of Cycle 10/10 using XGBoost	19
4.8	First approach - Accuracy and F1 score of the 10 Cycles test using XGBoost . . .	20
4.9	First approach - Accuracy and F1 score of the 50 Cycles test using XGBoost . . .	20
4.10	First approach - Accuracy and F1 score of the 100 Cycles test using XGBoost . .	21
4.11	Second approach - Mean difference, Accuracy and F1 score of the 10 Cycles test using Random Forest	21
4.12	Second approach - Mean difference, Accuracy and F1 score of the 20 Cycles test using Random Forest	22
4.13	Second approach - Mean difference, Accuracy and F1 score of the 50 Cycles test using Random Forest	22
4.14	Second approach - Mean difference, Accuracy and F1 score of the 100 Cycles test using Random Forest	23
4.15	Second approach - Mean difference, Accuracy and F1 score of the 10 Cycles test using XGBoost	23
4.16	Second approach - Mean difference, Accuracy and F1 score of the 20 Cycles test using XGBoost	24
4.17	Second approach - Mean difference, Accuracy and F1 score of the 50 Cycles test using XGBoost	24
4.18	Second approach - Mean difference, Accuracy and F1 score of the 100 Cycles test using XGBoost	25

List of Tables

3.1	Number of samples of Plain Weave and Satin Weave	12
3.2	Number of labels Berger_bin of Plain Weave and Satin Weave samples	13

Abbreviations

AI	Artificial Intelligence
LIME	Local Interpretable Model-Agnostic Explanations
ML	Machine Learning
MD	Mean Difference
RF	Random Forest
SHAP	Shapley Additive Explanations
SMOTE	Synthetic Minority Oversampling Technique
SMOTE-NC	Synthetic Minority Over-sampling Technique for Nominal and Continuous
XAI	Explainable Artificial Intelligence
XGB	XGBoost

Chapter 1

Introduction

This chapter will introduce the problem being proposed, the context in which it is inserted, the difference it can make in the industry of smart manufacturing, the goals for the current work and a simple structure of the present document.

1.1 Context

Artificial Intelligence has been growing exponentially over the years, especially in the last decade. In just ten years, engineers developed systems that are able to detect diseases based solely on imaging medical exams, voice recognition devices to transform speech to text or to assist us in our day-to-day lives, cars that drive autonomously with barely any human interaction, and automatizing decision systems such as health insurance providence or candidates curriculum screening in the Human Resources departments of companies. Regarding the latter intelligent systems, it has become of extreme importance to understand the decisions behind those models, usually in the law and insurance areas, to ensure every human is treated equally regardless of their gender, race, sexual orientation or economic situation. Explainability is helpful in computing the variables that led to certain decisions and is crucial to detect and mitigating issues such as bias. However, this need to guarantee bias is controlled in an intelligent system can also be applied in a manufacturing environment, for example in cases of defect detection.

1.2 Motivation

Detecting bias is a major matter in Explainable AI algorithms today. With the broad adoption of AI in practically any system we could think of, especially those that directly or indirectly affect humans, any small favoring or prejudice can have a massive impact. This problem has been addressed in areas directly involving people, namely the financial or healthcare areas, where physical characteristics should have no weight on the final decision of a system. On the other

hand, in manufacturing, there is very little information on the topic. Seeing that this area has a multitude of different types of equipment, machinery brands and processes, detecting units with faults, tracing back the reason for these flaws and any tendency towards a specific defect could represent a great advantage for any manufacturing company. Plus, with the ongoing climate crisis, lack of natural resources and extreme levels of pollution, any opportunity to decrease waste, energy and time is invaluable, also representing a way of reducing expenses and increasing the profit margins.

1.3 Objectives and Research Questions

This work will focus on studying the effects of bias in data-based machine learning models. There are situations where it is not possible to correct bias while still keeping the model's accuracy above reasonable levels, as it's not clear if the tendency is in fact bias or just a feature of the dataset important for the final decision. Therefore, the main question to be answered is "Can bias be manipulated in data-based models related to smart manufacturing?"

Data-based models are used to make decisions, and these decisions will generate a new dataset, that will then be used to retrain the model. If bias is present in the first dataset used, every time the model is retrained the bias can be amplified, to a point where it will be spread out. In this case, the portion of the dataset that causes the bias will grow and eventually become the majority of the dataset, thus turning the bias into the norm. This effect will be analyzed using a biased manufacturing process dataset and named as "magnification effect". Then, a solution to detect bias in a defect prediction system will be developed based on this effect, and an approach to correct it in the early stages will be suggested so that its consequences are minimized.

1.4 Document Structure

In the present document, Chapter 2 starts by reviewing the literature that will be helpful and essential during the work development. Chapter 3 details the methodology and implementation used along the project, and Chapter 4 the experiments that took place and their results. Finally, Chapter 5 describes the conclusions and a suggestion for the next steps to take to improve the work done.

Chapter 2

Literature Review

In this chapter, the Literature Review on Explainable AI and Bias will be presented. The concepts are defined in a brief way and some methods to mitigate problems related to this dissertation are listed.

2.1 Explainable AI

Explainable artificial intelligence, commonly referenced as Explainable AI or XAI, is a set of tools and methods developed to help humans understand and interpret results produced by machine learning algorithms. It allows us to assess a model's transparency and fairness by tracing the path the system took to make a certain decision. Explainable AI is of utmost importance for those who develop data-based models, to ensure the development is carried out in a responsible way, by ensuring the model makes good decisions and thus increasing confidence in the system and building trust in its outcomes [24].

2.1.1 White-box and Black-box models

In regard to interpretability, ML models can be classified in two main groups: white-box models and black-box models. The first class consists of simpler models that can be easily understood by a human user, such as linear regression and decision trees. These models usually provide less predictive accuracy and as they not always can mirror the features' interdependencies and the complexity of the dataset, which makes it easier for someone with base knowledge in AI to trace back the model's outcomes. On the other hand, black-box models are very complex models, namely support vector machines and neural networks, that are virtually impossible to be interpreted by humans, even by experts in this field. In return, they present increased performance and prediction accuracy compared to white-box models [18].

2.1.2 Methods

The biggest challenge in Explainable AI is to make it simple to understand black-box models and to comprehend the outcomes they produce. To accomplish that, there are several well-recognized model-agnostic approaches (which do not depend on the type of ML model being analyzed) that will be briefly described in this section.

2.1.2.1 Local Interpretable Model-Agnostic Explanations

This approach, generally named LIME, consists of breaking down a complex model into multiple local simple models that achieve the same results in the same space interval. These new models are named surrogate models as they act as substitutes for the original model, and can be easily understood, making it possible to understand the decisions of the complex model. The main logic of the method is, after selecting a sample from the input data, to select neighbor values, evaluate the original model in this interval, and produce a new surrogate model that is approximated to the original. Now that LIME has produced an interpretable model, we can explain decisions that were previously non-interpretable [23].

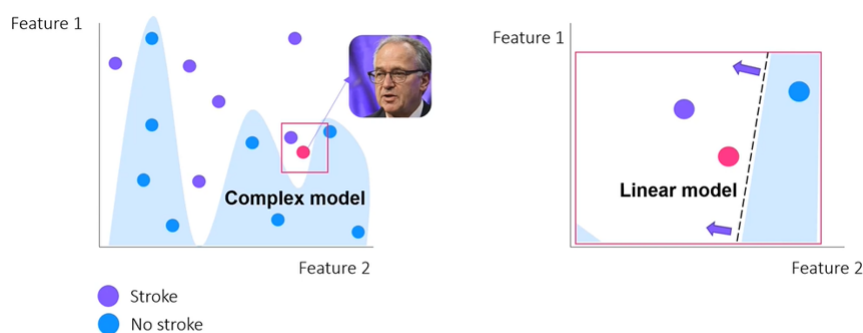


Figure 2.1: Example of complex model and LIME's surrogate model [5]

2.1.2.2 Shapley Additive Explanations

Shapley Additive Explanations, also known as SHAP, is a method based on Shapley values, a concept originally from game theory. Shapley values, in the context of explainability, represent the importance or weight contribution a feature has in the final output of an ML model. The goal of SHAP is to compute the Shapley values of all features for a certain chosen sample, and, if needed, for all samples of the dataset, to then aggregate and create a global contribution analysis. To assess the importance of a feature in a model, SHAP calculates the results of the model with and without a single feature and compares the data to determine the exact contribution. However, as some features only show their real weight in the final outcome when paired with other features, SHAP also creates combinatory subsets of features and compares the output with and without said subgroups. This way, the true contribution of a feature can be exactly computed [19].

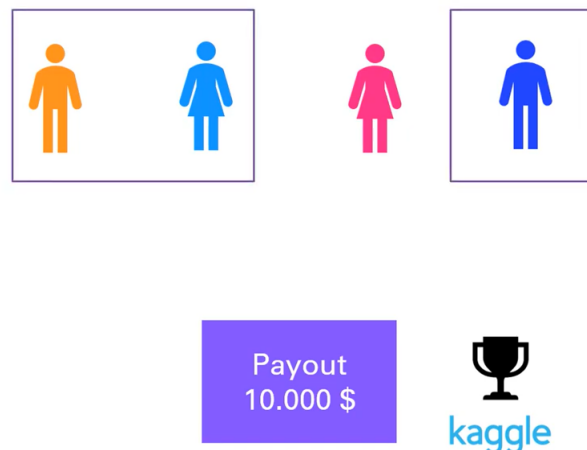


Figure 2.2: Example of SHAP subsets, in the context of a prize division within a group of people [6]

2.2 Bias

Bias is one of the major problems and concerns Explainable AI tries to mitigate. It is, by definition, any favoring or prejudice towards a specific group in a way that is not fair when compared to all other groups. It can be a hard task to detect bias, as we have to be extremely careful when managing data, to distinguish what is in fact bias from what could be valuable information for the system to use and learn from. Such a problem involves a great mastery of the business knowledge of the model and that is another reason why this task is of such hard execution.

2.2.1 Detecting Bias

In order to assess the level of bias in a dataset, there are currently several techniques used in the ML field that can be applied. These methods can range from determining the impact of protected features in the target variable to absolute measures that are calculated to evaluate the extent of differences between two or more groups of data, generally between the most favored group and the remaining ones.

2.2.1.1 Feature impact calculation

One of the most common options to analyze bias is by quantifying the relative effects of the input features on a model's predictions. This method can be used to assess the model's fairness as the relative importance of each feature on the model's output is calculated and measured through a percentage number. Using state-of-the-art algorithms that assess feature impacts such as LIME or SHAP, explained in more detail in Section 2.1.2, is a way of checking if protected features that should not have a significant influence on the prediction are impacting the model in any way. In case of a protected feature having a weighty impact in comparison with other input variables,

that influence should be further investigated and the feature at issue should not be included in the model.

2.2.1.2 Mean difference

The mean difference [10], or statistical parity difference, quantifies the difference between the mean value of the target variable of the protected group and the general group.

$$d = E(y^+|s^0) - E(y^+|s^1)$$

If the difference is zero or negligible, then it can be considered that the dataset is not privileging any group.

2.2.1.3 Impact ratio

The impact ratio [1] is the ratio of the number of positive outcomes for the protected group over the unprotected or general group.

$$r = p(y^+|s^1) - p(y^+|s^0)$$

The closer the ratio is to 100%, the less bias is present. This metric is used in the US courts as a way of measuring discrimination, with decisions being deemed as discriminatory if the value is below 80%.

2.2.1.4 Disparate impact

The disparate impact [7] is the ratio of the probability of a positive outcome between the unprivileged and the privileged groups.

$$r = \frac{p(y^+|s^0)}{p(y^+|s^1)}$$

When the value is 1, neither group is being favored; with values less than 1, the privileged group is being benefited, and with values above 1, the unprivileged group has some advantage.

2.2.1.5 Odds ratio

The odds ratio [21] is used to estimate the association between exposure and outcome, and it is related to the logistic regression. The exponential function of the logistic regression coefficient corresponds to one unit increase in the odds ratio.

$$r = \frac{p(y^+|s^0)p(y^-|s^1)}{p(y^+|s^1)p(y^-|s^0)}$$

2.2.2 Minimizing Bias

Bellamy et al. [2] gathered multiple algorithms to mitigate bias, grouping them into three sets: pre-processing algorithms, in-processing algorithms and post-processing algorithms [1]. In the first group, the algorithm runs before the model is trained and modifies the training dataset. These can include weighing each combination of groups and labels differently or reducing the dataset to ensure a more balanced distribution of groups. Regarding in-processing algorithms, the original model is adjusted so that it can better learn from the input during the training phase, for example by creating a classifier that reduces an adversary's capacity to determine the feature that is increasing the bias. In post-processing algorithms, the model is considered a black-box and therefore the only possible way to mitigate bias is after the prediction phase. This could be done by assigning positive outcomes to prejudiced groups and negative outputs to favorable groups.

2.2.2.1 Reweighting

Kamiran et al. (2012) [13] implemented the Reweighting method, a non-intrusive solution that assigns weights for training samples in each group and outcome combination, instead of the more common approach of modifying the samples' labels, such as in Reject option classification (Section 2.2.2.6). For example, samples from the unprotected group and a positive outcome will get higher weights than samples from the unprotected group and a negative outcome, while in the protected group samples with positive outcomes will have lower weights compared to the ones with negative outcomes. To calculate the weights, the ratio between the expected probability of seeing a sample in each group and the outcome combination and the observed probability is used. The sensitive attribute and the outcome have to be statistically independent in an unbiased dataset, D , so, for example, the expected probability for a sample in the protected group and with a positive outcome would be:

$$P_{exp}(S = 1 \wedge y = +) = P(S = 1) \times P(y = +) = \frac{|X(S) = 1|}{|D|} \times \frac{|X(y) = +|}{|D|}, \forall X \in D$$

However, the observed probability for the same sample is:

$$P_{obs}(S = 1 \wedge y = +) = \frac{|X(S) = 1 \wedge X(y) = +|}{|D|}, \forall X \in D$$

This translates into the weight being:

$$W(X) = \frac{P_{exp}(S = X(S) \wedge y = X(y))}{P_{obs}(S = X(S) \wedge y = X(y))}$$

2.2.2.2 Learning fair representations

Zemel et al. (2012) [25] proposed a preprocessing model that creates a representation of the original dataset that doesn't hold the sensitive attribute while keeping the rest of the information. This is done by mapping each sample, represented as a data point in the original space, to a

probability distribution in a new representation space. During the process of learning the new mapping, this solution also aims to maximize the accuracy of classification tasks using this new representation.

One crucial goal of the algorithm is to ensure statistical parity, which implies that the probability of a sample, k , from the protected group (+) of the original dataset, X , maps to a certain sample in the new representation space, Z , is equal to the probability of a sample from the unprotected group (-) maps to the same sample:

$$P(Z = k|x^+ \in X^+) = P(Z = k|x^- \in X^-), \forall k$$

Apart from satisfying statistical parity, this method also aims to retain all information except the sensitive attribute in the new mapped space, as well as keeping the classification decisions as close to original as possible.

2.2.2.3 Adversarial debiasing

Zhang et al. (2018) [26] presented an inprocessing algorithm, that runs during the training of the prediction model, and aims to maximize the accuracy of the system while reducing an adversary model's capacity of predicting the sensitive attribute from the prediction previously generated.

This solution is based on Generative Adversarial Network (GAN) and is divided into two steps: the predictor, where the dataset is trained to predict the original task of the model; and the adversary, which uses the output layer of the predictor as input to another network that aims to predict the protected attribute, and is equivalent to the discriminator step in a classic GAN. The adversary network can, however, have other inputs, depending on the definition of bias being used:

- Demographic Parity: uses the predicted label as input, which translates to the adversary having to predict the protected attribute based solely on the predicted label.
- Equality of Odds: uses both the predicted label and the true label.
- Equality of Opportunity: uses only samples with a certain label to train the adversary.

2.2.2.4 Prejudice remover

Kamishima et al. (2012) [16] introduced a technique that uses a fairness regularizer to minimize indirect prejudice, which is a statistical dependence between the protected attribute and the target label.

The regularizer attempts to reduce the prejudice index, a metric used to quantify indirect prejudice. For a dataset D , with target variable y and sensitive attribute s :

$$PI = \sum_{(y,s) \in D} \hat{P}_r[y,s] \ln \frac{\hat{P}_r[y,s]}{\hat{P}_r[y]\hat{P}_r[s]}$$

Marginalization of the model M over the non-sensitive features, X , is applied, as well as some approximations to increase efficiency in case of a large dataset, which results in the final prejudice remover regularizer $R_{PR}(D, \Theta)$, where Θ is the set of model parameters:

$$\sum_{(x_i, s_i) \in D} \sum_{y \in \{0,1\}} M[y|x_i, s_i; \Theta] \ln \frac{\hat{P}_r[y, s_i]}{\hat{P}_r[y]}$$

2.2.2.5 Equalized odds post-processing

Hardt et al. (2016) [11] proposed a post-processing solution to achieve equalized odds in classification problems by finding probabilities to change output labels.

In order to do so, a derived predictor that only depends on the original prediction and the protected attribute is implemented. The goal is to create a high-performing predictor, which quality can be assessed through the use of a loss function that returns the cost of predicting a wrong label. Therefore, a good equalized odds algorithm always aims to minimize the respective loss function.

2.2.2.6 Reject option classification

Kamiran et al. (2012) [14] suggested a method that increases the favorable outcomes of unprivileged groups and negative outcomes for privileged groups in a confidence band around the decision boundary with the highest uncertainty.

Using a single classifier, a posterior probability of having a favorable or unfavorable outcome is calculated. In case this probability is close to 0 or 1, the outcome produced by the original model is kept. If, however, the probability is closer to 0.5, there is more uncertainty regarding the sample and therefore it falls in the critical region. In this case, the outcome is swapped from the original model: if originally favorable, it's changed to unfavorable, and vice-versa.

Chapter 3

Methodology and Implementation

This chapter details the highlighted problem and the implementation used in the project, including different approaches to simulate a retraining process and two datasets.

3.1 Problem Definition

Machine Learning models require data to learn from and build knowledge about, which then applies to predict an output variable from a set of input data. Data is, however, far from perfect and mostly never covers all possible use cases, which is why it needs to be analyzed and curated before using it to train an ML model. Even then, a certain bias can be present in the samples: a tendency to favor or prejudice a set of data compared to the remaining samples.

In a manufacturing setting, as well as in other cases, ML models are continuously being improved to keep up with new data, enhance outliers coverage and improve overall accuracy. One method of doing so is to retrain the model with the predictions it produces when it is in active use, allowing the model to increase its knowledge but also contributing to reinforcing its weaknesses, such as bias, turning them into an expanding liability.

This work tries to recreate a manufacturing retraining ML process and evaluate the evolution of bias along the continuous training of the model, as well as suggest potential solutions to tackle this issue.

3.2 Datasets

For the development of this project, two datasets were selected: one related to defect prediction in a plastic film production line (Section 3.2.1) and another related to fabric whiteness levels in a dyeing process (Section 3.2.2).

3.2.1 Plastic Film Defects

The Plastic Film Defect Prediction dataset was retrieved from Kaggle¹ and consists of information regarding a plastic film production line from granular raw material.

The dataset comprises 226,536 samples, which is a considerable amount of data, each with 469 numerical features containing miscellaneous information about the production process: multiple temperatures and pressure levels, machine and materials dimensions, working times, and other settings from the production machines.

3.2.1.1 Data Cleanup and Analysis

The next step was to clean the dataset and remove all samples with null or infinite values, as they don't represent valuable data in this use case, and to ensure the results' quality was as high as possible. By applying this filter, the number of samples was reduced to 225,434, down 1,102 from the original dataset.

Originally, the data doesn't have a target variable for the defect representation, so, to achieve that, a new column was added based on the thickness of the final product: in case the thickness is equal to zero, the film broke during the production process and therefore a defect is present and the target variable takes value 1; if the thickness is more than zero, then the sample is free of defects and the target is 0.

Furthermore, a correlation analysis was executed to evaluate the relationship between features and avoid redundancy of information. Because the dataset had a high number of features, all variables with a correlation higher than 80% were removed, resulting in 268 features in the final dataset, 201 less than the initial data.

3.2.2 Fabric White Dyeing

The fabric dataset was sourced by DIGI2 Industrial Lab and comprises data about the process of white dyeing fabrics, which can be of two types of textile weave: plain weave and satin weave. The continuous target variable is *Berger*, a whiteness index that represents the white level of the fabric after applying the dye.

This dataset contains 154 samples, each with 13 numeric and categorical features such as the quantity of Caustic agent, Peroxide, Sequestering agent, and other agents in the dyeing solution, Roller pressure, Roller speed, and the type of textile weave.

3.2.2.1 Data Cleanup and Analysis

Initially, the feature "Fabric Construction" is a string value that represents the type of weave of the sample: either "Plain Weave" or "Satin Weave". To ease the training and predicting process, as well as the measurement of bias, the feature was hot-encoded, resulting in a new "Plain Weave"

¹<https://www.kaggle.com/datasets/podsyp/find-a-defect-in-the-production-extrusion-line>

feature, with a value of 1 for Plain Weave samples and 0 for Satin Weave samples. Each type of textile weave had the exact same number of samples, as shown in Table 3.1.

Table 3.1: Number of samples of Plain Weave and Satin Weave

Type of Textile Weave	Number of samples
Plain Weave	77
Satin Weave	77

As denoted in Chapter 2, most metrics and correction algorithms related to bias only apply to classification problems with a categorical feature as input. In this dataset, the target variable, "Berger", is a numerical feature, a positive float number with values ranging from 47.8 to 72.5. In order to transform it into a binary variable, "Berger_bin", a threshold of value 65 was set, based on information gathered from the manufacturer that provided the data. This translated to 34 samples above the threshold, having 1 as a value, and the remaining 120 having the value 0.

The dataset was also assessed to check if there were any null or infinite values, but the data was quite clean already and all values were valid. A correlation analysis was performed as well, and none of the features had a high enough correlation to be removed from the dataset, as can be seen in Figure 3.1.

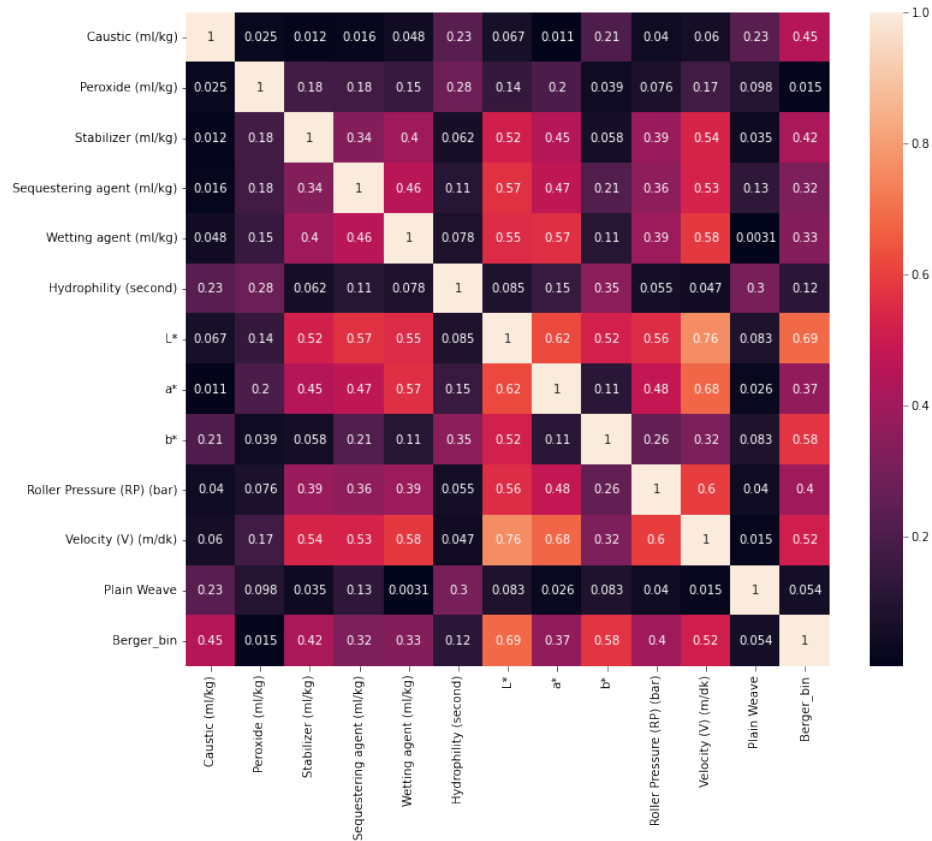


Figure 3.1: Correlation matrix for the Fabric White Dyeing dataset

Furthermore, the bias of the dataset was analyzed using the mean difference metric. This metric is the difference between mean target values of the protected and unprotected group, in this case, the difference between the mean of "Berger_bin" of "Plain Weave" and "Satin Weave".

Table 3.2 shows the number of samples with positive and negative labels in each category of textile weave. As shown, the numbers are very similar, and therefore it is expected the mean difference will result in a low value. After executing the algorithm, this theory proves to be true, with the dataset having a mean difference value of 0,026, which means there is a slight tendency towards a positive value in Plain Weave samples.

Table 3.2: Number of labels Berger_bin of Plain Weave and Satin Weave samples

	Berger_bin	
	1	0
Plain Weave	18	59
Satin Weave	16	61

However, it was clear that the reduced size of the dataset would represent a problem in the training and testing tasks, as there would not be enough samples to recreate the retraining process, detailed in Section 3.3, over time.

To tackle this issue, an over-sampling technique, Synthetic Minority Oversampling Technique (SMOTE) [4], was applied. SMOTE is an over-sampling method that generates new samples from a category of the dataset, usually the minority of a feature, by calculating the K-Nearest-Neighbour samples and creating a new sample from a random point between the selected samples. Synthetic Minority Over-sampling Technique for Nominal and Continuous (SMOTE-NC) [20] is a variation of SMOTE applicable for datasets with both categorical and numerical features, and therefore the appropriate version for this dataset. Although this technique is usually intended to equalize imbalanced datasets, it can also be utilized as a data augmentation method.

By using SMOTE-NC, it was possible to define how many new samples we wanted to generate, by defining that number as an argument, as well as which features are categorical and numerical. In order to have enough samples in a high number of retraining cycles, 10,000 new samples were generated in total, 7,912 with label 0 and 2,242 with label 1, keeping the same proportion of the original dataset.

3.3 Retraining simulation Approaches

In order to mimic the retraining process we were trying to simulate as described in Section 3.1, two approaches were implemented.

3.3.1 First Approach

The first method consists of splitting a dataset into N equal parts so that the training and testing process would be run N times. The goal is to, in each cycle, retrain the model with its previous predictions and with new samples, test it, save the predictions and evaluate the bias and performance.

Figure 3.2 shows a general flow of the process.

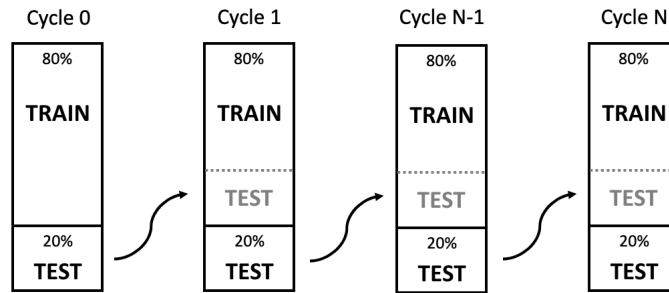


Figure 3.2: First Approach diagram

In the first cycle, the dataset is split randomly into training and testing subsets with sizes of 80% and 20% respectively and thereafter normalized. After preprocessing the data, the model is trained using one of the following ML classification models: Random Forest or XGBoost. These models were chosen as they are relatively time-efficient ML models and present overall good results for the time they take training and testing. The RF model is initialized with 100 estimators and property "warm_start" set to true, in order to allow the retraining of the model later and add more estimators to the ensemble, without overriding the previous forest. The XGBoost classifier is set with the objective "multi:softmax" (suitable for classification problems), 2 output classes (0 and 1), 100 estimators, a learning rate of 0.1 and the ensemble from the previous cycle to iterate on. The model is then trained using the training dataset and tested the first time against the testing dataset. After finishing the first testing task, the testing dataset's labels are swapped with the predictions generated by the model, creating a new dataset that will be used for the retraining in the next cycle. Next, the models are tested once again, now with a set of selected and constant samples from the original datasets (without added noise or generated samples, and that was not present in either the training and testing datasets of each cycle) to assess how the bias results and the model's performance would vary over the cycles.

In the following cycles, the process repeats itself with some slight changes. After normalizing the data, the predicted dataset saved previously is appended to the current training dataset, allowing the ML model to train with predictions it generated in a previous cycle. In the case of an RF classifier, 100 estimators are added to the model before fitting so that there are new Decision trees to be added to the forest. When using the XGBoost classifier, the previously fitted model is added as a property to the fitting method in order to train over the ensemble instead of creating a new one.

In the initial step of development of this project, while working with the Plastic Film dataset, every time the model ran the accuracy would always be 100%, which is an unrealistic value and indicates that the model was overfitted. Therefore, Gaussian Noise was added to the testing dataset in each cycle, after the normalization task, with 0 as the mean of the normal distribution and 0.1 as the standard deviation. The number of samples generated is the same as the number of samples in the testing dataset, making it double in size after adding the new samples.

3.3.2 Second Approach

The second method was only applied to the Fabric dataset and consists of two main tasks: the first one exclusively for initial training of the ML model and another for both predicting and retraining. The process is shown in Figure 3.3.

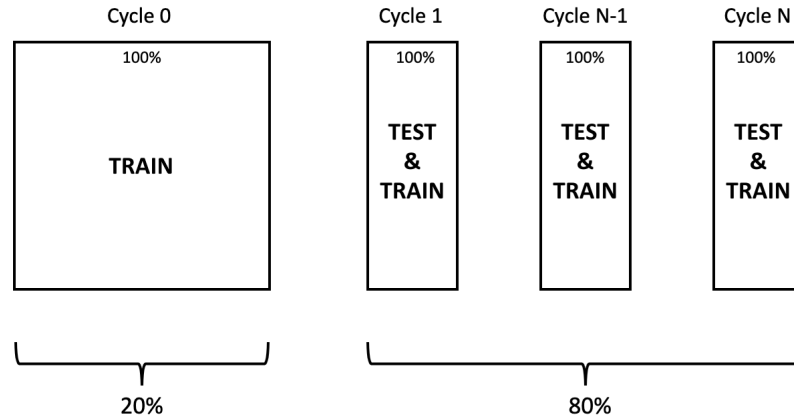


Figure 3.3: Second Approach diagram

For the initial training and following predicting and retraining tasks, the data used are all 10,000 samples generated using the SMOTE-NC algorithm, split for every cycle and therefore with new data in each task. To evaluate the performance and assess the model's bias, the original dataset of 154 samples is utilized to analyze how the model's predictions vary along the retraining cycles using the same input data. Firstly, the dataset is split into two datasets: 20% for an initial training task and 80% for the following retraining cycles. The next step is to train the model using the training subset, to simulate a model's training before it would be deployed in production. After that, the second subset is divided into N parts to test and retrain the model N times and replicate the data from the manufacturing processing line that would be used as input in the already trained ML model in production. In each cycle, the dataset is normalized and the ML model is tested against it. The models used are, as in the first approach, Random Forest and XGBoost, with the same configurations. The predictions generated by the model replace the labels in the dataset, and this new dataset created is then used to retrain the model. Next, the model is tested again, this time with the original dataset, and several metrics related to performance and bias are executed and saved for later analysis.

Chapter 4

Experiments and Results

This chapter presents the experiments that were executed on both datasets using the two approaches described in Section 3.3 and the respective results. It also mentions some setbacks that occurred during the development and suggestions to improve the results further.

4.1 Plastic Film Defects

4.1.1 Experiments

As previously mentioned in Section 3.2.1, the Plastic Film dataset did not have any categorical feature to allow us to create groups within the data and compare them in order to analyze if there was bias, *i.e.*, if one of them was being favored. It was also not possible to define a threshold in one of the features as we did not have the necessary business knowledge about the data (plastic film production), so any threshold that could be established could not be guaranteed of any meaning in the context of the production line.

Without a categorical feature to divide the dataset into multiple subsets (one for each value of category), one way of measuring bias would be by computing the feature's contribution weight to the prediction of the model, using one of the Explainable AI algorithms listed in Section 2.1.2. It was decided to apply SHAP, as it usually tends to obtain more accurate results than LIME while not being much more computationally expensive. Besides bias, the model's accuracy was also stored and analyzed.

This approach was executed multiple times with different configurations: 10, 50, and 100 cycles, and using a Random Forest or XGBoost model.

4.1.2 Results

When using the Random Forest Classifier, the results obtained from the SHAP algorithm didn't vary depending on the number of cycle runs. The 5 features that contributed the most to the final prediction were always constant, sometimes switching places between them without any observable pattern or tendency. Figures 4.1 and 4.2 show the 10 highest SHAP values (representing

each feature contribution) of different cycles, to give an example of how the features with the most importance remained the same, although in a different order.

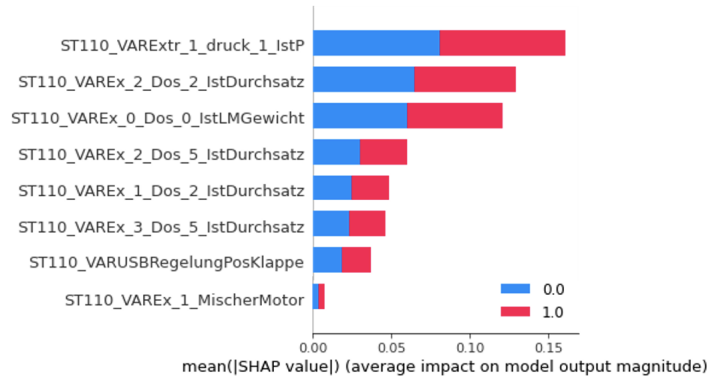


Figure 4.1: First approach - SHAP values of Cycle 1/50 using Random Forest

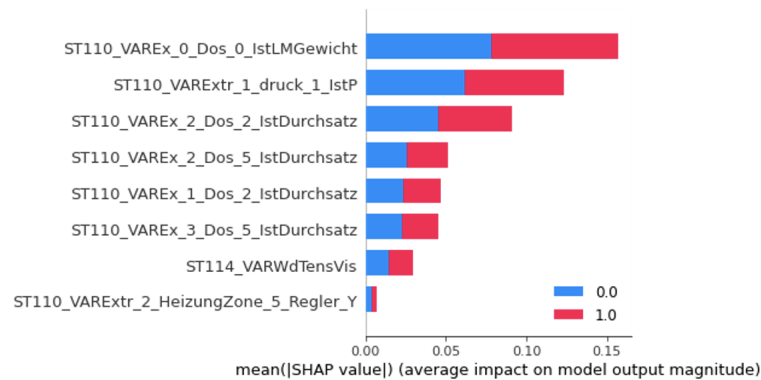


Figure 4.2: First approach - SHAP values of Cycle 10/50 using Random Forest

The model's performance was also assessed along 10, 50 and 100 cycles. On the 10 cycles test, accuracy remained constant throughout the cycles at around 72% and F1 had inconsistent variations, ranging from 31% to 52%, as seen in Figure 4.3.

During the 50 cycles experiment, accuracy dropped from 68% to 53% on the first 10 cycles, and then had only slight variations until the end. F1 score had, once again, volatile values, as shown in Figure 4.4.

When running this approach with 100 cycles, the performance was similar to those from the 50 cycles test: accuracy dropped initially by around 10% and F1 showed no pattern in its variation (Figure 4.5).

On the other hand, the experiments executed with the XGBoost Classifier produced rather different results. The feature "ST110_VAREx_0_Dos_0_IstLMGewicht", representing the extruder meter weight, consistently stood out as the one with the most impact on the model's prediction in the tests with 10, 50 and 100 cycles, as it can be seen in Figures 4.6 and 4.7.

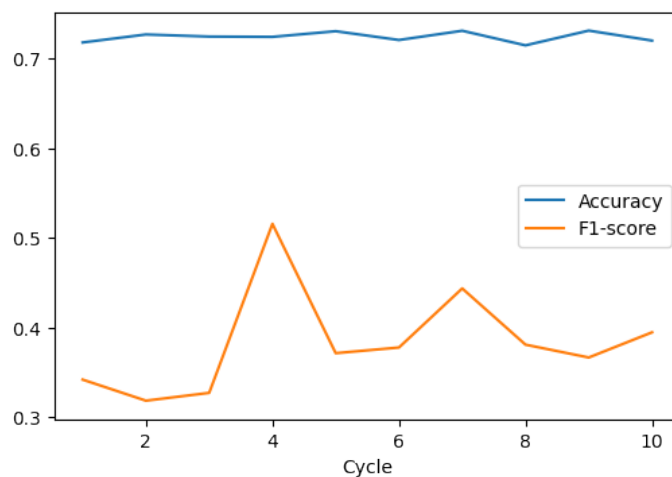


Figure 4.3: First approach - Accuracy and F1 score of the 10 Cycles test using Random Forest

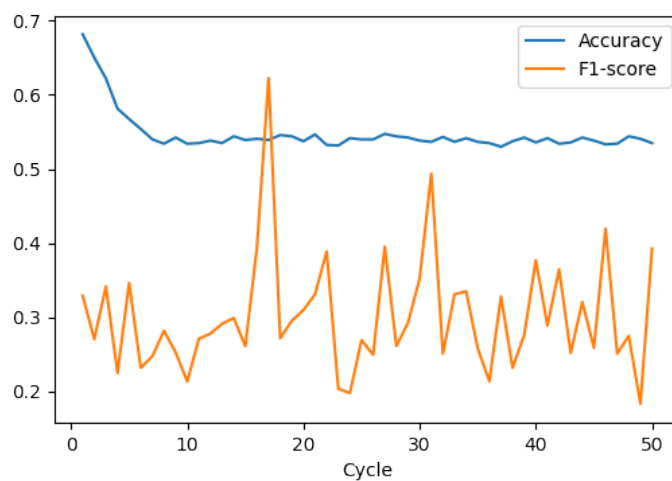


Figure 4.4: First approach - Accuracy and F1 score of the 50 Cycles test using Random Forest

Regarding the performance using XGBoost, the model had overall higher accuracy but similar F1 score values. On the 10 cycles test, accuracy stayed very consistent at around 74% and F1 had similar values compared to the RF model, as shown in Figure 4.8.

In the 50 cycles experiment, the model was able to sustain its accuracy at around 70%, contrary to the RF model whose accuracy dropped to 50%. F1 score presented the same random variation, as it can be seen in Figure 4.9.

At last, regarding the 100 cycles experiment, the model kept a relatively consistent accuracy of around 70% throughout the cycles, and, once again, very fluctuating F1 values, ranging from 8% to 78%, as illustrated in Figure 4.10.

When analysing the SHAP values of all the experiments, and without any business knowledge about the process of plastic film production, no conclusions could be reached regarding the variation of the model's bias using this dataset. The features with the most contribution are all numerical, and without insights about this manufacturing process no thresholds could be defined.

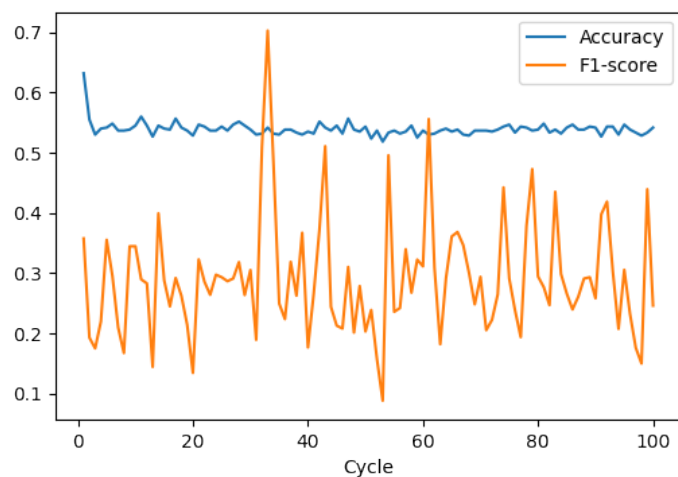


Figure 4.5: First approach - Accuracy and F1 score of the 100 Cycles test using Random Forest

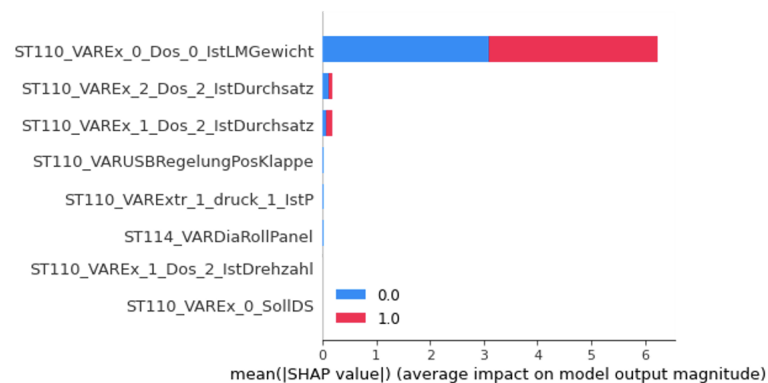


Figure 4.6: First approach - SHAP values of Cycle 1/10 using XGBoost

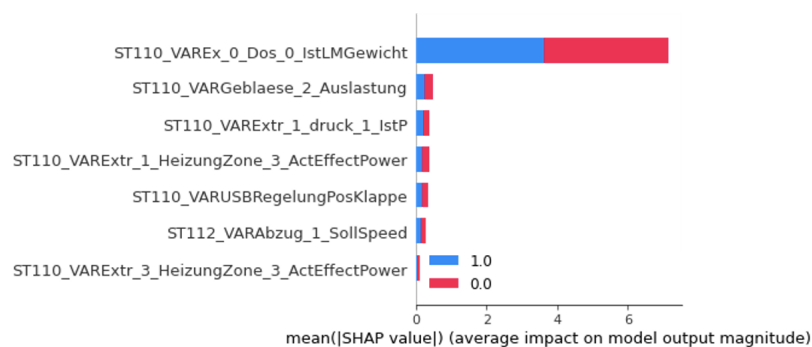


Figure 4.7: First approach - SHAP values of Cycle 10/10 using XGBoost

Furthermore, these features represent settings from the manufacturing machinery so it seems reasonable that they have a great impact on the final result and therefore can't be considered protected attributes.

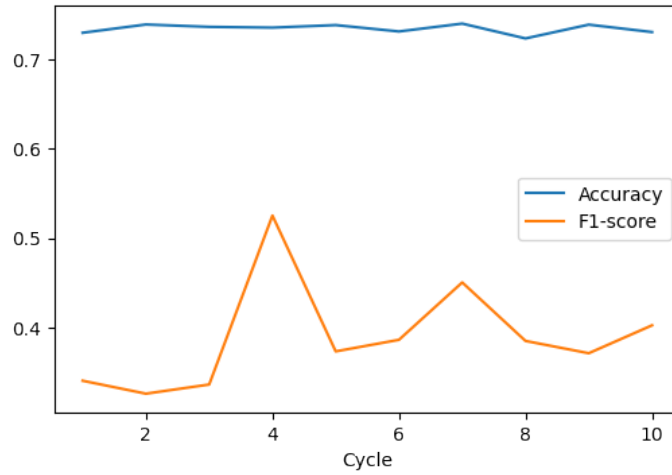


Figure 4.8: First approach - Accuracy and F1 score of the 10 Cycles test using XGBoost

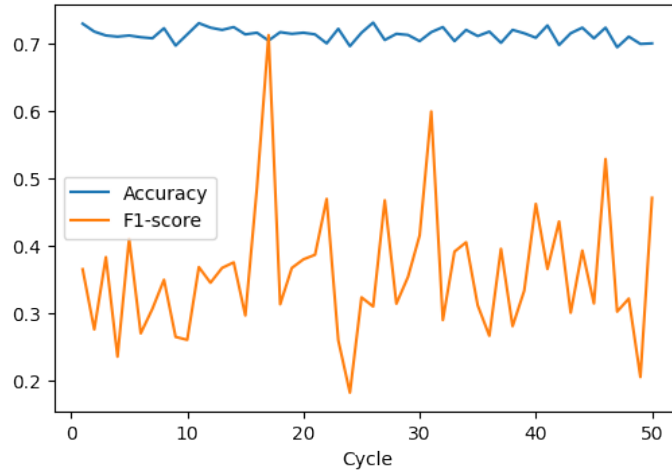


Figure 4.9: First approach - Accuracy and F1 score of the 50 Cycles test using XGBoost

4.2 Fabric Dyeing

4.2.1 Experiments

Using the Fabric Dyeing dataset, the Second approach described in Section 3.3.2 was tested. Experiments were conducted with 10, 20, 50 and 100 number of cycles, and using a Random Forest or XGBoost classifier, as in the first approach. To evaluate the bias of the model, the mean difference metric was chosen as it is widely used in the industry when assessing bias in a classification model between two groups of data. In this case, there isn't a specific privileged and unprivileged group, so it was arbitrarily chosen that the "Plain Weave" samples would be considered the privileged ones, and the "Satin Weave" the unprivileged. The model's accuracy and F1-score were also registered to check its performance.

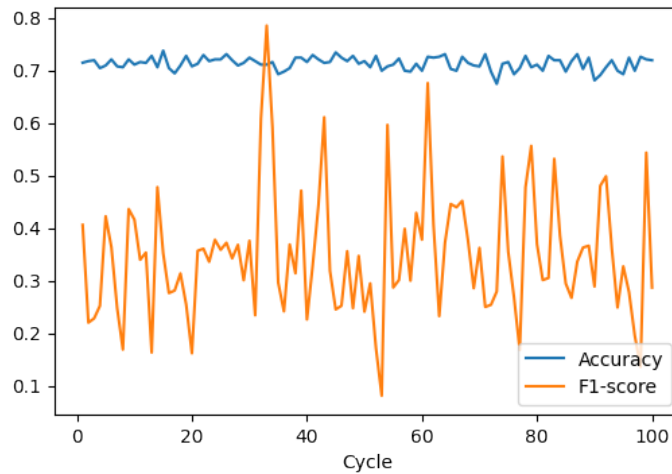


Figure 4.10: First approach - Accuracy and F1 score of the 100 Cycles test using XGBoost

4.2.2 Results

In the experiments using the Random Forest model, the mean difference metric showed an increase over time and the bias magnification effect could be proven.

When running the approach with 10 cycles, mean difference rises from 0,02 to around 0,78 in the first 4 cycles, showing a clear tendency towards the "Plain Weave" samples, and then stabilizes in the latter value, as seen in Figure 4.11. On the other hand, accuracy drops from around 95% to 53% in the same cycle frame, showing how the two metrics are correlated.

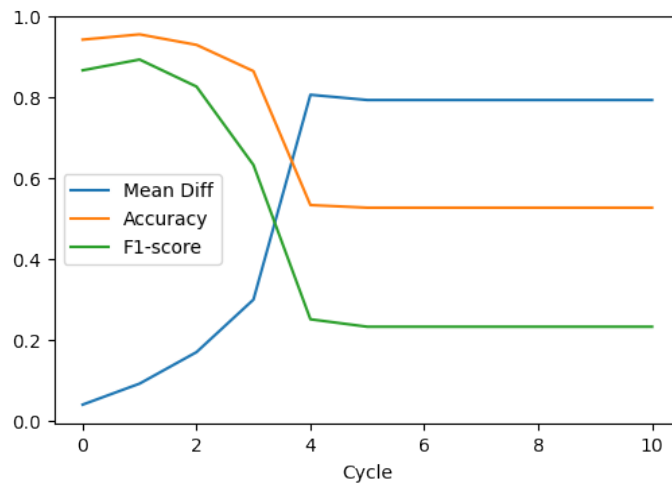


Figure 4.11: Second approach - Mean difference, Accuracy and F1 score of the 10 Cycles test using Random Forest

In the experiment with 20 cycles, mean difference has a substantial increase until cycle 5, where it reaches 0,80, which is maintained until the end of the test. Contrastingly, accuracy declines from around 95% to 53%, as illustrated in Figure 4.12.

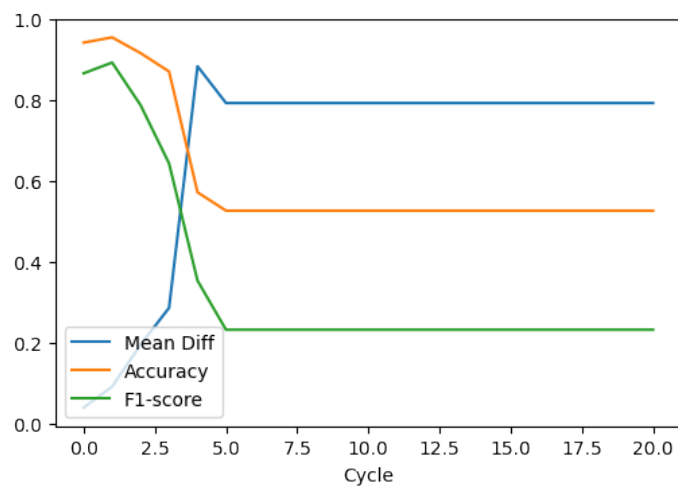


Figure 4.12: Second approach - Mean difference, Accuracy and F1 score of the 20 Cycles test using Random Forest

During the 50 and 100 cycles experiment, the behavior of mean difference and accuracy is similar, as shown in Figures 4.13 and 4.14. Mean difference rises up to 0,80 in the first 9 cycles, and accuracy drops to around 53% and 57%, in the 50 and 100 cycles respectively. However, in the last 10 cycles of the 100 cycles test, there is an 0,08 drop in mean difference and 5% in accuracy, which may indicate a degradation of the model's performance with a high number of retraining tasks.

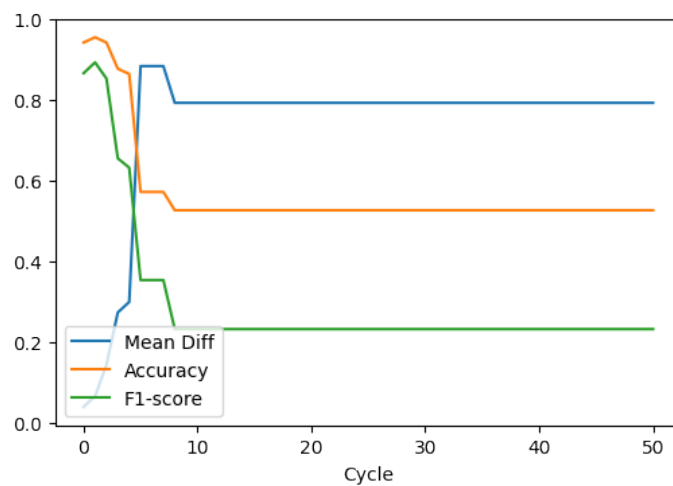


Figure 4.13: Second approach - Mean difference, Accuracy and F1 score of the 50 Cycles test using Random Forest

The experiments using the XGBoost classifier had very different and poorer results. Mean difference did not increase in any of the tests and accuracy did not achieve satisfactory results. As shown in Figures 4.15, 4.16, 4.17 and 4.18, mean difference had a value of 0 over the course of all the experiments, and accuracy dropped significantly in the first cycle to around 22%.

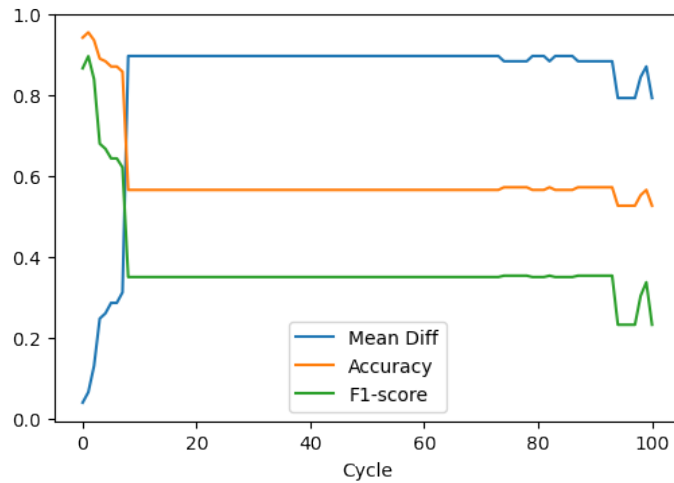


Figure 4.14: Second approach - Mean difference, Accuracy and F1 score of the 100 Cycles test using Random Forest

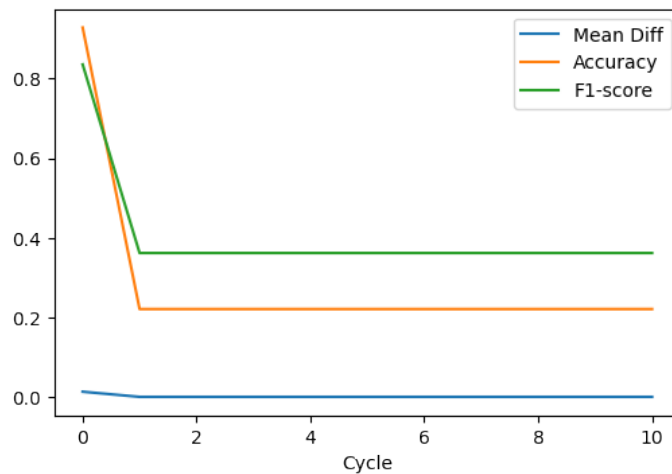


Figure 4.15: Second approach - Mean difference, Accuracy and F1 score of the 10 Cycles test using XGBoost

4.3 Summary

The experiments executed using the First approach didn't produce relevant results. The Plastic Fabric dataset doesn't have a categorical feature which makes the process of analyzing bias much more difficult, especially without having any previous knowledge about this specific manufacturing process.

However, with the Second approach, it is possible to prove the magnification effect of bias in a retraining process. When using a Random Forest classifier, bias increases exponentially until it reaches its limit at around 0.8, which indicates that the model has a tendency toward samples of the group "Satin Weave". This issue could be prevented by applying a bias correction algorithm, such as Reweighting (Section 2.2.2.1) or Reject option classification (Section 2.2.2.6), in the initial

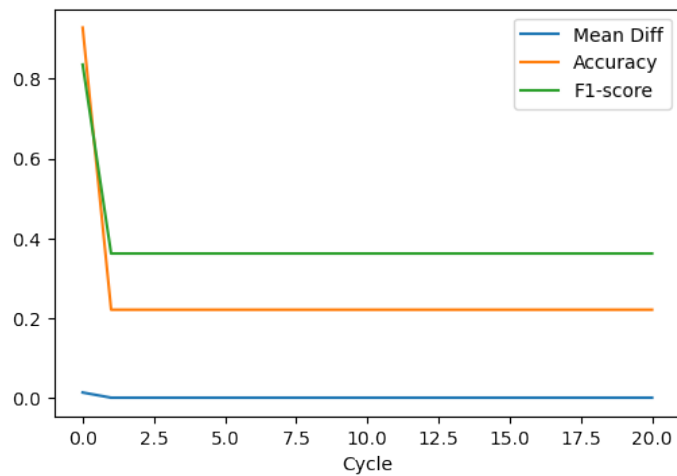


Figure 4.16: Second approach - Mean difference, Accuracy and F1 score of the 20 Cycles test using XGBoost

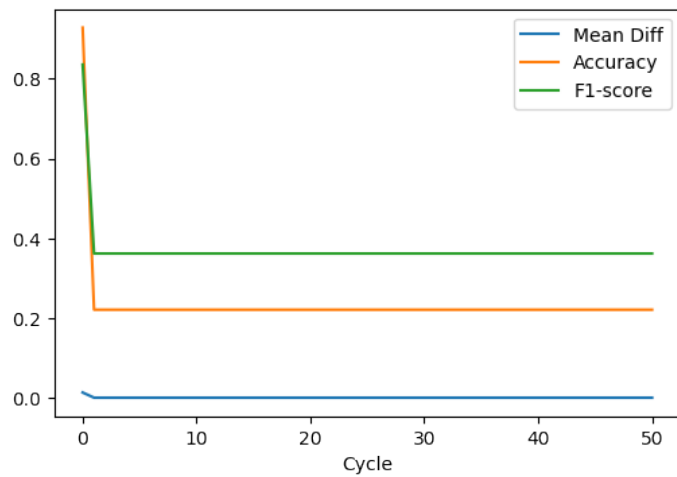


Figure 4.17: Second approach - Mean difference, Accuracy and F1 score of the 50 Cycles test using XGBoost

steps of training and even during the retraining tasks, when the mean difference value reaches a predefined threshold.

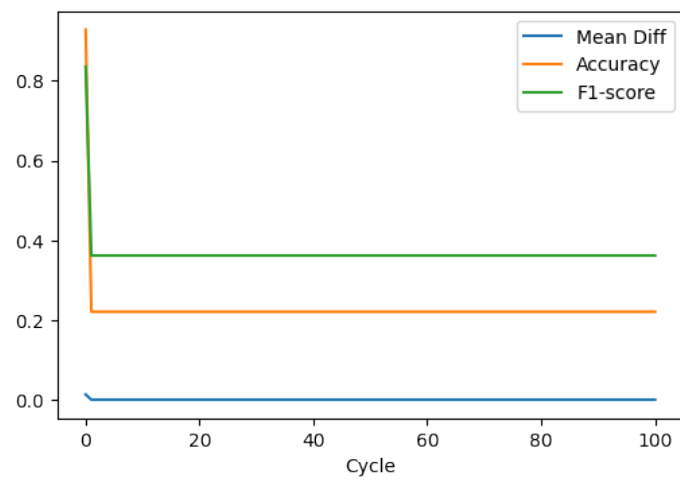


Figure 4.18: Second approach - Mean difference, Accuracy and F1 score of the 100 Cycles test using XGBoost

Chapter 5

Conclusions and Future Work

This chapter summarizes the research work and findings of this dissertation. The development and results achieved are described and a suggestion of next steps to take to continue exploring this topic is detailed.

5.1 Conclusions

Bias plays a very important role in data-based ML models, affecting their performance and, in real-life scenarios, possibly having serious consequences for their unfair predictions. Hence, it's crucial to detect it, monitor it and, if possible, correct it. Although bias has been studied in human-related situations, such as credit card approval, it can also present a problem in the manufacturing scene. From machinery brands to types of raw materials, there are multiple situations where an ML model might have a tendency towards a certain class compared to another in a production line.

This dissertation focused on studying the effect of bias in a manufacturing setting, analyzing its behavior in a simulated retraining process, proving a magnification effect, and finding a method to reduce or avoid it. There are already multiple metrics to assess the bias of a model, namely the mean difference, which is one of the most commonly used in the fairness field and was employed in this work as well. Furthermore, some bias mitigation methods were gathered in order to be able to suggest a solution to correct bias in its early stages.

The development comprised the implementation of two approaches that aim to simulate an already-in-place prediction system that is retrained over time using its own prediction. The First approach consists of splitting a dataset into N equal parts, each one to be used in the N^{th} cycle of the method. In each cycle, the model is fitted with new data from the N^{th} subset and with the predictions generated in the previous cycle. Then, it predicts the labels from a part of the cycle's subset, and the bias and performance are calculated with a set of samples randomly selected from the original dataset. This approach was applied to the Plastic Film Defects dataset using a Random Forest Classifier and an XGBoost Classifier, with each model being executed with 10, 50 and 100 cycles. It didn't produce results that could lead to some conclusions about the bias of the model,

mainly because it doesn't have a categorical feature to allow for a split of the dataset into different groups and compare them to check if the prediction is biased towards any of them.

Another approach implemented consisted of splitting a dataset into two parts: one for initial training, and another one for retraining cycles. After fitting the model with the first dataset, the second one is split into N subsets, that will be used in the N cycles. In each cycle, the model predicts the labels for the samples in the corresponding dataset, then is fitted with the labels produced and after that it's tested with an initially defined constant set of samples from the original dataset, to assess the performance and bias of the model along the cycles. This approach was applied to the Fabric Dyeing dataset, again using a Random Forest Classifier and an XGBoost Classifier, with each model being executed with 10, 20, 50 and 100 cycles. Contrary to the First approach, using the RF model the Second approach produced results that make clear that bias is increasing over the cycles until it stabilizes to a limit value, and thus proving the magnification effect hypothesized in the proposal of this dissertation, as well as the effect of bias in the performance of data-based ML models.

5.2 Future Work

Although the results presented in Section 4.2 met one of the goals of this thesis, there is still plenty of room for improvement and exploration.

The next step of development would be to apply a bias correction algorithm, one of those detailed in Section 2.2.2, for example. Different approaches could be implemented: apply these post-processing algorithms once only after the initial training, apply them in every cycle where bias surpassed a predefined threshold or a mixture of both. Another suggestion would be to execute a pre-processing algorithm to the dataset being used in the initial training and analyze how the bias would behave over time. One could also choose more computationally expensive methods, such as in-processing ones, to be run every time the model is fitted. In case bias is still increasing after applying one of the previous suggestions, it could also be considered to implement a combination of bias mitigation methods, such as pre-processing and post-processing, for example.

References

- [1] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Majsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John 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, October 2018.
- [2] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Majsilovic, 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*, 2018.
- [3] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
- [4] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- [5] DeepFindr. Explainable AI explained! | #3 lime. Available at <https://www.youtube.com/watch?v=d6j6bofhj2M>, Accessed last time in March 2022, 2021.
- [6] DeepFindr. Explainable AI explained! | #4 shap. Available at <https://www.youtube.com/watch?v=9haIOplEIGM>, Accessed last time in March 2022, 2021.
- [7] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’15, page 259–268, New York, NY, USA, 2015. Association for Computing Machinery.
- [8] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’15, page 259–268, New York, NY, USA, 2015. Association for Computing Machinery.
- [9] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th*

- ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, page 259–268, New York, NY, USA, 2015. Association for Computing Machinery.
- [10] Pratyush Garg, John Villasenor, and Virginia Foggo. Fairness metrics: A comparative analysis. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 3662–3666, 2020.
 - [11] Moritz Hardt, Eric Price, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.
 - [12] Bahrudin Hrnjica and Selver Softic. Explainable AI in Manufacturing: A Predictive Maintenance Case Study. In *Advances in Production Management Systems. Towards Smart and Digital Manufacturing*, pages 66–73, Cham, 2020. Springer International Publishing.
 - [13] Faisal Kamiran and Toon Calders. Data preprocessing techniques for classification without discrimination. *Knowl Inf Syst*, 2012.
 - [14] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *2012 IEEE 12th International Conference on Data Mining*, pages 924–929, 2012.
 - [15] Faisal Kamiran, Asim Karim, and Xiangliang Zhang. Decision theory for discrimination-aware classification. In *2012 IEEE 12th International Conference on Data Mining*, pages 924–929, 2012.
 - [16] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 35–50, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
 - [17] Guanpeng Li, Siva Kumar Sastry Hari, Michael Sullivan, Timothy Tsai, Karthik Pattabiraman, Joel Emer, and Stephen W. Keckler. Understanding error propagation in deep learning neural network (DNN) accelerators and applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, New York, NY, USA, 2017. Association for Computing Machinery.
 - [18] Octavio Loyola-González. Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. *IEEE Access*, 7:154096–154113, 2019.
 - [19] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *CoRR*, abs/1705.07874, 2017.
 - [20] Mimi Mukherjee and Matloob Khushi. Smote-enc: A novel smote-based method to generate synthetic data for nominal and continuous features. *Applied System Innovation*, 4(1), 2021.
 - [21] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. On fairness and calibration. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
 - [22] Amifa Raj and Michael D. Ekstrand. Measuring fairness in ranked results: An analytical and empirical comparison. In *Proceedings of the 45th International ACM SIGIR Conference on*

- Research and Development in Information Retrieval*, SIGIR '22, page 726–736, New York, NY, USA, 2022. Association for Computing Machinery.
- [23] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016.
- [24] Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer Nature, 2019.
- [25] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 325–333, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.
- [26] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. *CoRR*, abs/1801.07593, 2018.