Robustness and Observational Fairness in Dynamic Environments

[Extended Abstract]

December, 2023

Tiago Pinto
tiagofrancopinto@tecnico.ulisboa.pt

Advisors: Prof. Dr. Mário Figueiredo, Dr. Pedro Saleiro and Sérgio Jesus

***Abstract*—Machine Learning (ML) is starting to find its way into a broader range of applications, including high-stakes decision-making settings such as finance and healthcare, where errors can heavily impact individuals' lives. The added responsibility in these scenarios requires thorough testing to guarantee correct model behavior after deployment. Consequently, robustness testing emerges as a core aspect of the development process. This study focuses on financial fraud detection, a dynamic landscape replete with a myriad of perturbations that obstruct the model's intended behavior, making robustness analysis a difficult yet indispensable endeavor. Given the high-stakes implications of the ML application, decisions profoundly influence individuals' lives, emphasizing the necessity of ensuring that algorithms do not discriminate based on factors such as gender, race, and the like. Hence, our approach bridges the realms of robustness, dynamic environments, and fairness, introducing a novel perspective on robustness that accounts for the specific perturbation under scrutiny, considering both performance and fairness. Guided by this vision, we present a taxonomy of perturbations, ranging from data-related issues to adversarial attacks, and suggest a framework tailored to evaluate the robustness of models against these perturbations. Subsequently, we put this framework into action by conducting tests on three distinct tabular datasets, applying those perturbations, and exposing popular models' lack of robustness. We also build a model with robustness concerns and employ our framework to demonstrate that it is more robust than its predecessor.**

***Index Terms*—adversarial attacks, algorithmic fairness, dynamic environments, machine learning, robustness, tabular data**

I. Introduction

Artificial intelligence (AI) and Machine Learning (ML) are becoming a central tool in society. Their usefulness is growing with the amount of data available and the increasing computational power of our devices. The range of applications for this technology is no longer limited to simple tasks such as movie recommendations, but it is now also used in high-stakes decision scenarios. Said scenarios can be, for example, in financial services [1], in healthcare [2], and criminal justice [3].

ML algorithms frequently operate within dynamic and ever-shifting environments, replete with a myriad of perturbations that pose a considerable challenge to their ability to maintain their intended behavior [4]. In this context, the need for thorough testing becomes evident, ensuring that models do not experience substantial performance degradation in the face of these perturbations [5]. Particularly within high-stakes settings, where errors bear a deep influence on individuals' lives, an additional layer of scrutiny is indispensable. Consequently, robustness testing emerges as a core aspect of the development process [6], [7].

One of the applications where machine learning is thriving is financial fraud detection (e.g., detection of money laundering, scams, and impersonation). Correctly distinguishing a legitimate action from a fraudulent one as fast and accurately as possible is a high-value asset in the financial industry, resulting in large monetary investments in solutions for this task [8]. This is a high-stakes ML application where fairness is of the utmost importance, as people's lives are directly affected by the predictions, and should not be discriminated against based on sensitive protected attributes such as race or gender. Moreover, fraud detection is highly dynamic and adversarial, featuring complex interactions between the models and the environments, namely perturbations that affect the statistics of the data. These perturbations can range from natural distribution changes [9] to adversarial attacks [10] to data-related issues [11].

Therefore, ML models to detect fraud must not only be performant and fair but also robust to adversarial attacks and time-changing conditions. Although resource-expensive, retraining the model with the most recently available data is still the most common approach to deal with the challenges of dynamic environments [12]. Another option is to develop robust models that are able to resist those perturbations, thus avoiding the retraining process.

Even though performance is known to be affected by the perturbations characteristic of dynamic environments [9]–[11], their effect on fairness is still largely unexplored. Our goal is to make progress in this direction, studying robustness and observational fairness in dynamic environments. A significant part of our work will focus, not only on how perturbations affect the performance of ML models but also on its impact on fairness. To this end, we apply different perturbations to three tabular datasets and evaluate their robustness regarding fairness and robustness.

II. Background and Related Work

A. Observational Fairness

Fairness refers to how impartial a system is. Intuitively, fair decisions cannot be based on special protected categories of personal data, e.g., sensitive attributes such as gender or race. However, there is no universally accepted mathematical definition for measuring fairness. Simply removing the sensitive attributes is not the optimal solution, because other features can serve as a *proxy* for said attribute (as they are correlated with them), and may even lead to a decrease in both performance and fairness [13]. To better understand algorithmic unfairness, we need to understand the different metrics used for measuring unfairness and their relationship to distinct principles of fairness.

In this work, we will focus on the observational fairness metrics outlined by Barocas et al. [13], where the criteria are properties of the joint distribution of the features X, the score R, the sensitive attribute A, and the target variable Y. Broadly speaking, all fairness metrics equalize some group-dependent statistical quantity across groups characterized by some sensitive feature A. Simplifying the notion of fairness, criteria can be divided into three fundamental groups, each trying to equalize a different statistical quantity, according to 3 different principles:

Principle 1. Independence: this first criterion requires the sensitive attribute to be statistically independent of the score:

$$R \perp\!\!\!\perp A. \tag{1}$$

Principle 2. Separation: this next principle addresses the complaints regarding the previous one, as there is a difference between accepting a positive and a negative instance. By using the target variable Y, we divide the data into strata of the equal claim to acceptance, giving us a sense of *merit*. To satisfy these considerations, we need to guarantee conditional independence within each of these strata

$$R \perp\!\!\!\perp A \,|\, Y. \tag{2}$$

Principle 3. Sufficiency: The third and final condition is sufficiency, which assumes that the score R already takes into account the sensitive feature A to predict the target variable:

$$Y \perp\!\!\!\perp A \,|\, R. \tag{3}$$

In this work, we use the separation-based criteria. Although they still have their limitations, we exclude metrics based on principles 1 and 3, due to their disregard for the true value of the target variable and because sufficiency usually comes for free thanks to common ML practices (enforcing it is not much of an intervention), respectively. The metrics based on separation try to equalize the error rates $\mathbb{P}(\hat{Y} = 0|Y = 1)$ and $\mathbb{P}(\hat{Y} = 1|Y = 0)$ of a classifier $\hat{Y}$. One of the metrics using principle 2 is equalized odds [14], which consists of enforcing the True Positive Rate (TPR) and False Positive Rate (FPR) to be equal for different sensitive groups a and b,

$$\mathbb{P}(\hat{Y} = 1|Y = 0, A = a) = \mathbb{P}(\hat{Y} = 1|Y = 0, A = b), \tag{4}$$

$$\mathbb{P}(\hat{Y} = 1|Y = 1, A = a) = \mathbb{P}(\hat{Y} = 1|Y = 1, A = b). \tag{5}$$

This metric can be relaxed into using only Equation 4 or 5, which represent predictive equality [15] and equality of opportunity [14], respectively.

One of the main goals of *fairML* is to increase fairness by sacrificing as little performance as possible, considering the fairness-performance tradeoff, as studied by Corbett et al. [16]. Furthermore, there are also tradeoffs among fairness metrics themselves. Numerous studies show that it is impossible to simultaneously satisfy multiple notions of fairness. For example, it has been shown that if a classifier score is calibrated and group-wise prevalences are different, there is no way to balance both FPR and FNR [15].

The *fairML* literature also proposes numerous methods for unfairness mitigation, including pre-processing, in-processing, and post-processing approaches [17].

B. Robustness and Dynamic Environments

Unlike static environments, dynamic ones feature complex relations among their elements which influence model behavior. This means that ML models deployed in those settings need to withstand the effects of such dynamics.

1) Robustness: A common approach to dealing with the undesired dynamics of the environment is to constantly retrain the model with more recent data. However, this process is costly and sometimes impossible, as in cases such as the fraud detection scenario, there is an inherent delay in the labeling process (often, a fraudulent instance takes a long time to be uncovered, leading to labeling uncertainty). To avoid this struggle, one should aim to build robust models. Robustness can be defined as the resilience of the system's correctness in the presence of perturbations [18]. The causes of the perturbations can be malicious intent (i.e., adversarial attacks) or of other nature (i.e., covariate shift due to consumer habit change). Regarding the correctness of the model, it can refer to various properties of the model, namely performance and fairness metrics. All in all, it means that robustness is tied to the ability of a model to maintain similar levels of correctness with and without the presence of invalid inputs or stressful environmental conditions.

Instead of focusing on evaluation, the field of robustness is heavily oriented toward the development and optimization of methods for adversarial attack generation. This leads to the small number of available metrics to measure robustness. Even so, some studies focus on quantifying robustness [19], [20].

2) Perturbations: ML models are trained using data that follows a specific distribution and, more often than not, that is not the same distribution of the data on which they have to predict in *production*. There is an array of perturbations that lead to this change in distributions.

The real world is a dynamic place and constantly changes. Natural occurrences might modify data distribution, making it differ from training to testing time - *distribution shift*. Among the distribution shifts studied in the literature, the two major categories addressed are *covariate shift* and *concept shift*.

Covariate shift refers to the distribution of the input features, or covariates, not being the same at training and testing time, but the same not happening to the conditional distribution of the outcomes given the features [21]–[23]. Conversely, *concept shift* is the scenario where the distribution of the input features stays the same, but the conditional distribution of the outcomes given the features changes [24], [25].

In decision-making scenarios, using a model to support the decision process, can trigger actions that influence the outcome it aims to predict. This phenomenon is called *performative prediction* and it is associated with a distribution change in the target data caused by the model itself [26]. Examples of *performative prediction* are *strategic classification* and *selective labels.* The first consists of anticipating individuals who may intentionally set their attributes in a configuration that leads to favorable classification [27]. The latter consists of situations where the model completely determines the label of an instance: for example, in fraud detection, rejecting one instance of an account opening, due to believing it is fraudulent, means that its true label will never be observed [28].

Another perturbation that ML models are subject to is *adversarial attacks*. *Adversarial attacks* are situations where an attacker intentionally feeds the model with misleading or incorrect data, thus causing the model to make mistakes or behave in an unwanted manner. The goal of the attacker might consist of compromising the learning process (*poisoning attacks*), evading the system at the testing time (*evasion attacks* or learning as much as possible about the model (*exploratory attacks*). [29]

Zooming in on the predominant attack category, *evasion attacks*, it is worth noting that these incursions can be grouped into distinct categories based on their capabilities during the testing phase. In this context, attacks can be classified as either *white-box* or *black-box* attacks. The first category assumes adversaries have total knowledge about the model used for classification and the second considers the attackers have no information about the model, making them less threatening than white-box attacks, but more realistic. Specifically, financial fraud detection can be considered a black-box hard label setting, where the adversary has no information about the model, thus only being able to query the model and assess the hard label output. [29] Examples of white-box attacks are DeepFool [30] and C&W attacks [7], while for black-box attacks we have Boundary Attack [31] and HopSkipJump Attack [32], for instance.

The most common approach to deal with this problem is using *adversarial training*. Generally speaking, adversarial training consists of a data augmentation technique that generates new adversarial samples from a set of victim examples [10], [33], [34].

3) Observational Fairness in Dynamic Environments: In dynamic environments, it may be harder to achieve fairness because the data and the context in which the models are being used can change. However, it is important to take a proactive approach to address these obstacles to ensure ML is being used in a non-discriminatory way. Given that, attention to fairness in dynamic environments has been growing. Some works show the impact of the aforementioned phenomenons on fairness and how to mitigate it.

If the distribution of the data changes from training to testing time, performance in different groups might be affected, thus influencing fairness. Studies regarding *distribution shift* [35], [36], *performative prediction* [37], [38], and *adversarial attacks* [39], [40] have shown the effects of these perturbations in fairness and proposed methods to mitigate them.

III. A Framework for Evaluating Robustness

FairML studies mostly focus on static environments. The topic of robustness is widely studied in the literature, but it is mostly connected to adversarial attacks and performance. Furthermore, the adversarial attacks are almost exclusively directed at image data. In this thesis, the task at hand is always binary classification using tabular data. Unlike images, this type of data is heterogeneous, featuring dense numerical features and sparse categorical features [41], providing a different set of challenges when developing models and crafting adversarial attacks. The objective of this work is to provide a robustness view that takes, not only performance but also fairness into consideration, as well as a framework to evaluate model robustness using tabular data.

A. Overview

In general supervised learning requires large amounts of labeled data. Data scientists gather available data and use supervised learning to obtain the best possible model according to some objective. For example, in a credit attribution setting, one can use a model to predict whether the customer will pay it back, and the data used to train these models are input-output pairs, where the input is a group of features (like housing status, occupation, etc...) and the output is a binary variable that signals if the credit was repaid. With this in mind, data scientists use their knowledge to build statistical models that, given a new input, aim to correctly predict the outcome. Using appropriate evaluation metrics, a model is finally obtained so that it can be deployed and used in the real world. Here, it is said to be in *production*, and its objective is to be useful to its users, meaning that, given new data, it needs to correctly predict the outcome. The lifecycle of a model is shown in Figure 1.

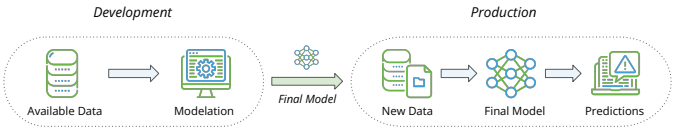

Fig. 1. Model lifecycle.

Data scientists develop statistical models using data that has a certain distribution which shapes the input-to-output mapping that is obtained. However, the world is dynamic and

complex, so the data that will be used in *production* rarely follows the same distribution as the one used in *development* [42]. Countless factors contribute to this phenomenon, namely data collecting problems, consumer habit change, and malicious intent, among others.

When the data distributions change, so does model behavior. Most commonly, the major problem that the entity that deploys the model worries about is performance degradation. However, with growing concerns about responsible AI, these entities should also account for its impact on fairness, especially if the models are used for decisions that have a great impact on individuals' lives. For instance, imagine a bank that operates in a certain country and develops a model that helps in a loan attribution setting, predicting if it will be paid back, using data gathered over the years on that location. The model is deployed and is a success, the bank thrives and begins expanding, opening a new branch in a new country on the other side of the world. If this bank wants to deploy its model in this new country, it needs to be wary that the demographics, data quality and reliability, and even the laws of the location might differ, thus leading to different data distributions. This may lead to unwanted model behavior, namely discrimination. Loan attribution is a high-stakes scenario and considering fairness concerns is vital, making it crucial to evaluate model robustness regarding fairness, not only performance.

With this in mind, we want to test how the models behave in the presence of these changes in distribution and evaluate their robustness, regarding not only performance but also fairness. To mimic this setting, we apply a set of perturbations to the data (thus simulating what happens in the real world) and study their impact on performance and fairness. A visual representation of our setup to mimic the real-world lifecycle is shown in Figure 2.

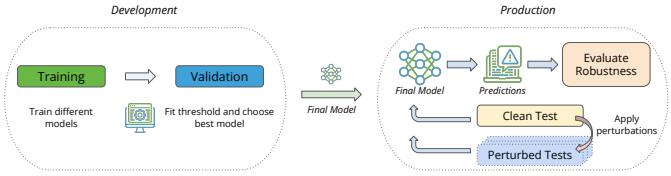

Fig. 2. Model lifecycle setup.

Our setup can be divided into 5 different steps: (1) split the available data into three different sets: training, validation and test; (2) use the training data to train multiple datasets; (3) use the validation set for choosing the best model - and tune a decision threshold, if necessary; (4) apply perturbations to the test set, as it mimics *production*, generating *perturbed test sets*; (5) evaluate model robustness using both clean and perturbed test sets.

These first models are developed without any robustness concerns, but it is evaluated nonetheless. Initially, we want to study how models are affected by these perturbations in case they are not taken into account in the model development process. After gathering these results, our focus moves towards developing robust models, so that we can lower the unwanted effects of the perturbations. Our approach for enhancing robustness consists of changing the training process, using *adversarial training* instead - the most common method used in the literature.

B. Perturbation Taxonomy

The real world is a dynamic and complex environment, making it difficult for a model to keep its correctness after deployment. Data continuously changes over time, affecting how models behave. These *shifts* can be caused by intangible reasons such as consumer habit changes, demographics, seasonal changes, and so forth. Our dynamic environments feature these shifts, however, the perturbations that we will enforce and include in our taxonomy are tangible ones: causes for shifts that we can observe and quantify.

It would be impossible to list every perturbation that a model can suffer in *production*. Given that our setting focuses on tabular data and, more specifically, on financial fraud, we can narrow them down to a more relevant subset. Taking this into consideration, we propose a taxonomy of perturbations for this study consisting of the most common problems that one has to face in this scenario; data issues and adversarial attacks. These challenges are very important when it comes to detecting financial fraud and they each come with their unique problems.

1) Data Issues: Data Issues refer to problems that harm the quality of the data. These issues can be caused by factors such as problems in data acquisition, and preprocessing done by entities, among others. This might lead to changes in data distribution. Loss of information, introduction of bias, and model instability are some consequences that may arise due to data issues. To better understand model behavior against these perturbations, we study the influence of three different perturbations within this category: feature dropping, feature corruption, and feature clipping.

Feature Dropping. In *production*, models make their predictions on new data. In an ideal scenario, every single instance of data would be filled, with no empty values. Unfortunately, that is unlikely. In most applications, some instances will have missing values, for various reasons, such as data acquisition sensors malfunctioning, incomplete surveys or forms, missing historical data, or changes in reporting standards.

Feature Corruption. Another problem that may arise is that, even though a certain value is present, it is meaningless - this is, corrupted. When applied in production, models might be using a corrupted feature in their predictions, possibly affecting their performance and fairness. Some reasons that might lead to data corruption are data entry errors, hardware failures, software bugs, and data transfer issues.

Feature Clipping. Instead of completely losing its value, a feature can be simply distorted. When making predictions in *production*, models need access to their data somehow. Often, data is provided by entities that process it beforehand. These entities might truncate or set lower and upper bounds for some features, altering their distribution.

2) Adversarial Attacks: Adversarial attacks are a category of challenges that pose significant threats to the robustness of ML models. These attacks can be initiated by malicious actors or adversaries aiming to deceive or manipulate the model's predictions, which is undoubtedly a topic of interest in fraud detection. Adversarial attacks can lead to deviations in the model's decision boundaries and undermine its ability to make accurate predictions. Consequences of adversarial attacks include increased vulnerability to fraudulent inputs, a decrease in model accuracy, and potential ethical concerns regarding model fairness and security [43]. In our setting, it makes sense to study *evasion attacks*, where the attacker targets the test set aiming to slightly perturb the input and fool the model.

Furthermore, to better mimic the process of adversarial attack generation in fraud detection, we assume a black-box decision-based setting, where the attacker has no information about the model. In this adversarial attack category, the attacker can only query the model with certain inputs and observe the hard-label binarized output. As the available attacks in the literature are developed for image classification tasks, we adapt two of them to a tabular data setting: Boundary Attack [31] and HopSkipJump Attack [32].

In this work, we want to apply our robustness evaluation framework, which considers fairness. We are aware that tabular data has diverse datatypes and different feature importances, and some of the fields are often not accessible, which influences how perpetrators forge their attacks, calling for the use of custom norms when measuring the size of perturbations. However, for the sake of simplicity and the possibility of testing multiple datasets (that would require multiple custom norms), we use the l_2 norm, which although does not represent a faithful representation of real-world applications, allows for testing our methods in a high-level approach. The aforementioned chosen attacks have already been proven effective in tabular settings and fraud detection datasets [44], so using them allows for focusing on our robustness framework. Using the l_2 norm means assuming all features are equally important for attack detection and accessible to the adversary.

Boundary Attack. The primary objective of this attack is to discover the minimal perturbation required to induce a misclassification. Boundary attacks are characterized by their iterative nature: starting with an initial input that is correctly classified, they systematically introduce small modifications while continuously monitoring the model's responses. Instead of the conventional gradient-based optimization approach used in convex problems, the Boundary Attack employs another strategy —specifically, a randomized exploration of the decision boundary. This approach navigates the boundary using a random walk, striving to identify the optimal perturbation for successful misclassification.

HopSkipJump Attack. The essence of this attack is the same as the Boundary Attack, differing only in the optimization scheme. Instead of using random walks on the boundary, this method changes the optimization problem so that it can use a gradient-based approach, making it more effective.

C. Evaluating Robustness

We previously defined model robustness as resilience against perturbations, meaning a model is robust if it maintains levels of correctness when perturbed. Our approach regarding robustness is based on this definition, thus a model is more robust the more it resists perturbations.

Robustness holds a central focus within this thesis, making it imperative to establish a means of evaluation. However, we argue that robustness should be tailored to the specific problem rather than relying on a universal metric. The ability of a model to withstand one type of perturbation, such as adversarial attacks, differs from its resilience to noisy data. Bearing this in mind, we consistently assess robustness in a manner most suited to the particular problem at hand, thus proposing a framework for evaluating robustness that takes into consideration the perturbation which the model's resilience is being tested against.

1) Robustness against Data Issues: To be robust against data issues, a model has to maintain levels of correctness (fairness and performance) when perturbed by them. Given that the injection of data issues into the *clean test set* results in various *perturbed test sets* (one for each perturbed feature, as they are perturbed separately), a robust model would be one that had similar levels of both fairness and performance on each *perturbed test set* and the *clean test set*. Taking this into account, the evaluation of robustness against these perturbations should be based on the difference in performance and fairness between clean and perturbed sets.

The robustness report of a model being evaluated against data issues consists then, of the differences of the values for the chosen metrics for the problem between *perturbed* and *clean test sets*. For example, if we are evaluating robustness against feature dropping in a setting with 5 features where the chosen evaluation metrics were accuracy and equality of opportunity, for performance and fairness respectively, we would have a difference in accuracy and a difference in equality of opportunity for each one of the 5 features.

2) Adversarial Robustness: Unlike data issues, adversarial attacks are purposely designed to make models misclassify samples. That said, if we let an adversary perform attacks unconstrained, every attack would flip model predictions. This means that if we attacked the whole test set, we would have an accuracy of 0%. This renders the previous approach for evaluating robustness useless, as values for performance would always be 0, and values for fairness would be unstable (ratios, for example, would be divisions by 0).

A possible solution for this problem would be to add constraints to the attacker so that the success rate wouldn't be 100%. However, this approach would not be problem-agnostic, thus sparking a need for tuning, making it less appealing. Instead, we propose measuring attack effectiveness, sweeping across different constraint values and observing the relations among them. The overall view of this approach is that the harder it is to forge effective attacks, the more robust a model is against adversarial attacks.

A successful adversarial attack is an attack that fools the model, leading it to misclassify the target sample. However, some properties make a successful adversarial attack more effective. An attack that requires fewer queries from the model to be created is more effective than one that needs more queries. Additionally, a smaller perturbation, according to some chosen norm, is more effective, as it results in an adversarial sampler more similar to the original one [7].

Thus, to evaluate performance robustness, we sweep across different values of the number of queries constraint and register the values obtained for perturbation norms (in successful attacks). Additionally, we fix the number of queries and sweep across various perturbation sizes as constraints, and observe how the success rate evolves. In the latter, an attack is only deemed successful if the perturbation behind it is smaller than the constraint.

Robustness regarding fairness calls for a slightly different approach. If an attack is more effective against one sensitive group than another, it means the model is less robust for that group. This leads to a need to learn how attacks affect each group individually. The metrics used to measure attack effectiveness are the same, but this time we need to take into account groupwise concerns. With this in mind, we present the same results for the perturbation size separately for each group.

Furthermore, we propose two new metrics for evaluating robustness regarding fairness. The first one represents the ratio between perturbation sizes for each group. We will consider the perturbation size measured as an l_p norm for simplicity (any custom norm can also be used).

$$PSR = \frac{\min(l_p^{g1}, l_p^{g2})}{\max(l_p^{g1}, l_p^{g2})}, \tag{6}$$

where l_p^{g1} denotes the norm for one sensitive group and l_p^{g2} for the other one. The Perturbation Size Ratio (PSR) is the ratio between the sizes, forcing the highest value to be on the denominator, and guaranteeing values in the [0,1] range. The other proposed metric consists of taking the ratio between the success rate (SR) for each group: the Sucess Rate Ratio (SRR) is computed as

$$SRR = \frac{\min(SR^{g1}, SR^{g2})}{\max(SR^{g1}, SR^{g2})}, \tag{7}$$

where SR^{g1} denotes the success rate for one sensitive group and SR^{g2} for the other one. This again forces the highest value to be in the denominator to keep values in the desired [0,1] range. These metrics are useful as they reveal whether a model is more robust for one sensitive group than the other.

IV. Robustness Against Perturbations

We applied the perturbations in three different scenarios, one of which was a dataset from the BAF Suite [45]. The Base Variant of the suite, or BAF Base as we denote it, is the dataset that recreates the original real-world online bank account opening fraud detection dataset. Each row corresponds to an application for opening a bank account. This dataset perfectly fits our intentions as it features distribution drift over time, both from natural occurrences (e.g. changes in client behavior) and fraudsters adapting to fool better the model, as well as biases that call for fairness concerns. We account for fairness regarding age, where we choose a group of 50-year-old people or older and another of people younger than 50 years old. As for the biases, there are group size and prevalence biases, with the older group having fewer instances and higher fraud prevalence.

We test 2 different ML algorithms: LightGBM (LGBM) [46] and feed-forward neural networks (NN). For each algorithm, we train 50 models on the training set using Random Search for hyperparameter optimization without robustness concerns and choose the best-performing one on the validation set. For evaluation metrics, we use TPR and Predictive Equality for performance and fairness, respectively.

This section reports the results obtained in the test set, which mimics the *production* phase of a machine learning model. Considering that robustness depends on the perturbation being studied, we evaluate model robustness against data issues and adversarial robustness separately.

A. Data Issues

To evaluate robustness against data issues, we need to first evaluate fairness and performance in the *clean test set*. Regarding the LGBM, we have TPR=56.5% and FPR Ratio=31.3%. As for the NN, we have TPR=49.3% and FPR Ratio=34.1%. Note fairness is low for both, as they were trained in a biased dataset.

In our taxonomy, we've seen 3 different categories of data issues, each with different implementations. The implementations consist of the following: dropping by replacing every instance's feature value with the mean (in case it is a numerical feature) or the mode (in case it is a categorical feature); corrupting by shuffling all the feature values; clipping by setting a threshold at the 10th and 90th percentiles and clipping every value to that range.

As for robustness evaluation, we bootstrap the test 50 times and compute the mean and 95% confidence intervals for the evaluation metrics. We've seen that robustness against data issues is measured by the differences in performance and fairness between clean and perturbed test sets. The differences for any given metric x, the mean and the confidence interval's lower and upper bounds, are computed as

$$\Delta\bar{x} = \bar{x}_{pert} - \bar{x}_{clean}, \tag{8}$$

$$\Delta x^{low} = x_{pert}^{low} - \bar{x}_{clean}, \tag{9}$$

$$\Delta x^{up} = x_{pert}^{up} - \bar{x}_{clean}. \tag{10}$$

These values allow for testing the variability of our method and evaluating the robustness of the model against each data issue applied. If the difference is negative, the lower it is, the less robust the model is, as it loses performance or fairness from *development* to *production*. On the other hand, if the

TABLE I
ROBUSTNESS EVALUATION OF THE MODELS TRAINED IN BAF BASE.

		Feature Drop - Mean/Mode		Feature Corruption		Feature Clip - 10/90	
		LGBM	NN	LGBM	NN	LGBM	NN
TPR	Perturbed Sets Created	29		29		24	
	Perturbed Sets w/ mean $<$ clean CI	14	12	12	9	4	2
	Average Delta	-4.33 pp	-1.69 pp	-1.89 pp	-1.55 pp	-0.56 pp	-0.46 pp
	Lowest Delta	-23.71 pp	-20.74 pp	-10.48 pp	-10.12 pp	-2.86 pp	-5.6 pp
FPR Ratio	Perturbed Sets Created	29		29		24	
	Perturbed Sets w/ mean $<$ clean CI	13	10	2	4	1	1
	Average Delta	-1.07 pp	-0.23 pp	0.35 pp	0.56 pp	-0.03 pp	0.07 pp
	Lowest Delta	-10.74 pp	-13.25 pp	-4.75 pp	-3.35 pp	-2.29 pp	-1.44 pp

difference is non-negative the model maintains (or increases) levels of correctness, meaning it is robust.

To better visualize the results, we present the results in Table I with the following information: number of *perturbed test sets* created (how many features were perturbed), number of those *perturbed test sets* where the mean value of the metric (each table will represent one metric) is outside the confidence interval of that same metric's confidence interval in the *clean test set*, under the lower bound, and also the mean and lowest values of the metric's mean value of all *perturbed test sets.*

Focusing on performance, the perturbation to which the models are the least robust is feature dropping, resulting in 14 and 12 TPR values falling below the *clean test set* confidence interval for LGBM and NN, respectively. For this specific perturbation, the models also exhibit their lowest values for both average and lowest deltas, registering at -4.33 percentage points and -23.71 percentage points, respectively, with both of these records attributed to LGBM. Feature corruption emerges as the subsequent perturbation with the most effectiveness, causing a decline in performance that falls below the confidence interval for 12 instances in the case of LGBM and 9 instances for NN. Additionally, the LGBM model records the lowest values for both the average delta and lowest delta, which are observed at -1.89 percentage points and -10.48 percentage points, respectively. As for feature clipping, it is the least impactful perturbation, affecting only 4 cases for LGBM and 2 cases for NN, with average delta values in proximity to zero percentage points.

Regarding fairness, robustness is notably low against feature dropping. Within this specific perturbation, a total of 13 instances for LGBM and 10 instances for NN deviated below the *clean test set* confidence interval, and a lowest delta of -10.74 percentage points was recorded for LGBM. These models exhibit greater resilience when confronted with the other two types of data issues. Specifically, LGBM's fairness metrics fall below the clean confidence interval in 2 instances of feature corruption and 1 instance of feature clipping. In the case of NN, these metrics drop below the clean confidence interval in 4 instances for feature corruption and 1 instance for feature clipping.

B. Adversarial Attacks

An adversarial attack, in fraud detection, corresponds to a fraudster trying to fool a model into not recognizing a fraud instance. In other words, it corresponds to a positive labeled instance that is predicted as negative by the model, a false negative. With this in mind, the instances that will be targeted to create adversarial attacks will be true positives, aiming to turn them into false negatives.

In our implementation, we target 1,000 true positives (or every true positive, in case there are fewer than 1,000) in the test set, to get reliable results about the effectiveness of our methods. We apply our attack algorithms to all of these instances. We use the default parameters defined by the authors and set a max query threshold of 10,000 queries. The attacks we apply, Boundary Attack and HopSkipJump Attack, were developed for image data, while the models here use tabular data for predictions. This means that we need to convert an instance into a numerical format (we one hot encode categorical features and scale every numerical feature to a range of [0, 1]) for applying a perturbation and then back to the original format before querying the model.

For evaluating robustness, we need information about each iteration of the attacking methods regarding the number of queries made and the current best l_2 norm. This way, we can sweep over them and compute the metrics we mentioned to describe the effectiveness of the adversarial attacks: success rate, l_2 norm (perturbation size), and number of queries.

The Figure 3 plots provide insights into adversarial robustness. Both attacks reach around $l_2 \approx 0.65$. The Boundary Attack is more effective than the HopSkipJump Attack against LGBM, but for neural nets, they're equally effective after 2,000 queries. Group-specific $l_2(queries)$ plots reveal slightly lower values for the old group, suggesting less model robustness. Notably, Perturbation Size Ratios (PSR) remain relatively stable, hovering near 1.0, despite l_2 differences between groups.

In Figure 4, we observe the success rate of adversarial attacks on dataset-trained models and the Success Rate Ratio (SRR). For LGBM, the Boundary Attack outperforms HopSkipJump, converging faster. Neural networks are also more vulnerable to the Boundary Attack below l_2=0.75, with similar

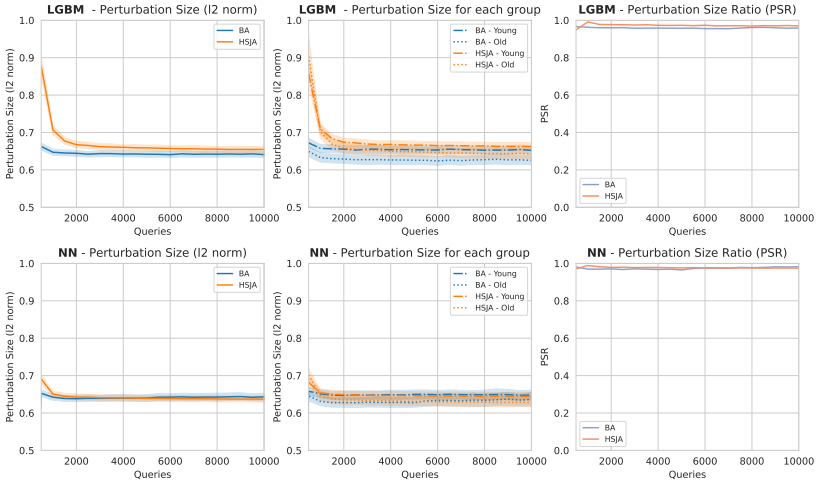

Fig. 3. Evaluating adversarial robustness regarding performance and fairness by measuring perturbation size (l_2 norm).

success rates beyond. Notably, a sudden success rate spike occurs around l_2=0.75, possibly linked to one-hot encoded categorical features. Concerning SRR, both models exhibit analogous patterns. After an initial spike (reaching ≈50%), the ratio stabilizes around 0.80. This suggests superior robustness for one group at perturbation with sizes $l_2 \leq 0.75$. After a second spike near l_2=0.75, the ratio approaches 1.0, signifying equal robustness at this perturbation size.

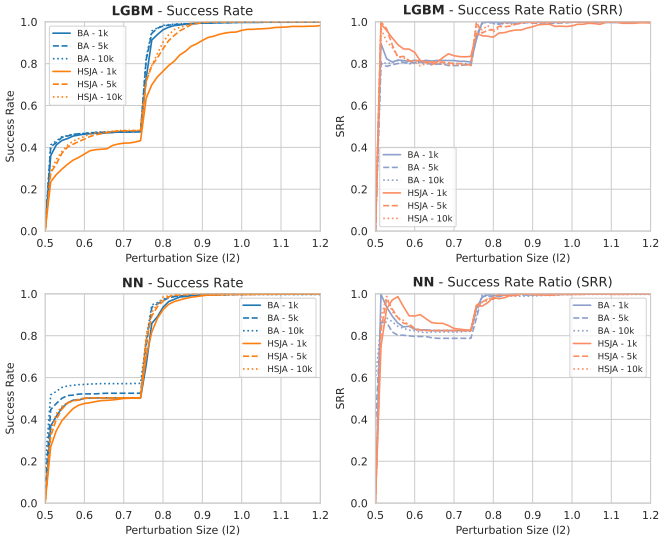

Fig. 4. Evaluating adversarial robustness, regarding performance and fairness, by measuring the success rate.

V. Enhancing Model Robustness

A. Overview

The objective of a robust model is to maintain both fairness and performance levels from *development* to *production*. These models account for the existence of changes in the data during the *development* phase, aiming to reduce their impact on the model's correctness. This way, a model's behavior in production should be closer to the one expected from development. Therefore, entities should be wary of methods for building robust models, preventing unwanted scenarios like the ones aforementioned.

When it comes to building robust models, the existing literature heavily leans towards Adversarial Robustness, which consists of the ability of a model's resilience against adversarial attacks. The literature provides numerous methods to increase adversarial robustness, and among them, adversarial training stands out as the most common. Adversarial training consists of crafting adversarial examples and incorporating them into the training process, exposing the model to these kinds of samples, so it can learn them. In our work, we choose to use this method for building robust models, as it is the most used approach.

The HopSkipJump Attack is selected for generating adversarial samples for training. This choice is driven by its similar effectiveness against LightGBM compared to the Boundary Attack, making it preferable for multi-round processes due to its better time complexity. Additionally, we set constraints with $l_2 \leq 0.8$ for perturbation size and a maximum of 5,000 queries to limit success rates. These values align with our previous findings: the norm is high enough for the success rate to be 100% in the first adversarial round and the number of queries is chosen at a point where the method has already converged. The model chosen is the LGBM, as it is the most common method used in this domain, and during the training process, model hyperparameters are kept the same as the ones obtained for the model in the previous section in every round.

B. Results

To assess the success of this process, we now turn to evaluating the model's resilience against the very same perturbation. We leverage the introduced framework, given that our primary aim was to boost the model's adversarial robustness, we focus solely on its evaluation in this specific context, as robustness is

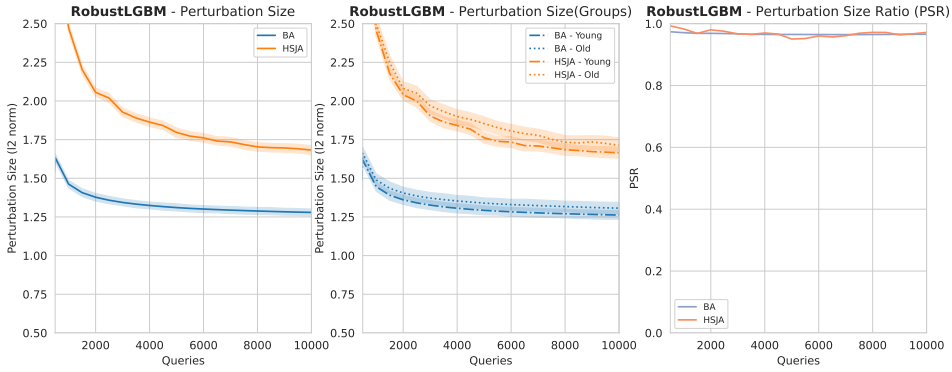

Fig. 5. Evaluating adversarial robustness of a model trained to be robust, regarding performance and fairness, by measuring perturbation size (l_2 norm)..

intricately tied to the nature of the perturbation. Even though the adversarial training process used the HopSkipJump Attack to generate the adversarial samples that were added to the training set, we also test the model against the Boundary Attack, given that it is still an adversarial attack. Results regarding perturbation size and PSR are shown in Figure 5.

Regarding fairness, PSR remains consistently close to 1.0, just like the model trained with no robustness concerns. RobustLGBM exhibits robustness across both old and young demographics, regardless of the attack type, with slightly higher robustness for the older group. Notably, the l_2 norms of attacks against RobustLGBM are more than twice those against standard LGBM, highlighting its increased robustness. Furthermore, while it significantly improves resilience against the HopSkipJump Attack, it also increases its ability to withstand Boundary Attacks.

In Figure 6, the results for the SR are shown. The evolution of SR progresses at a notably slower pace. For instance, the HopSkipJump Attack with 1,000 queries only reaches a 50% success rate at l_2=2.5, with the remaining attack-query combinations achieving 100% success rate near this point. In the realm of SRR, it undergoes a gradual convergence towards 1.0, with diminishing fluctuations across all attack-query scenarios as it approaches this value.

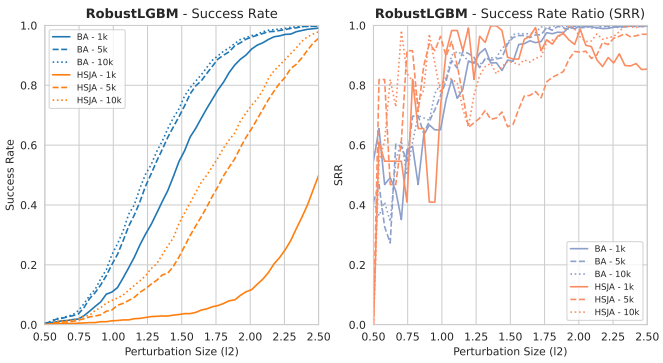

Fig. 6. Evaluating adversarial robustness of a model trained to be robust, regarding performance and fairness, by measuring the success rate.

VI. Conclusions

Our endeavor was to establish a fresh perspective on robustness, extending beyond adversarial robustness and encompassing fairness. We argue that robustness should be tailored to address distinct forms of perturbations. As a result, we propose a perturbation taxonomy that delineates various challenges that ML models must navigate to maintain correctness.

Furthermore, we introduce a framework that aligns with our robustness perspective, complementing the evaluation of models. This approach allows robustness to become a key consideration in the lifecycle of ML models in real-world applications. Running our fraud detection scenario, we employ an empirical analysis resorting to our framework to unveil the vulnerability of popular models that were constructed without robustness concerns. These settings are distinctive as they allow for testing with tabular data, particularly in a severely imbalanced context, and notably in high-stakes decision-making environments where fairness considerations are imperative. Additionally, we tested training with a focus on adversarial robustness with promising results when compared with standard model training.

VII. Future Work

For future work, we aim to broaden the perturbation taxonomy to include a wider array of challenges. For instance, we could introduce intangible perturbations, such as distribution shifts resulting from natural occurrences (e.g. shifts in consumer behavior) and dynamic biases.

We could also extend the taxonomy of perturbations in the topics of data issues and adversarial attacks. When it comes to data issues, we have the option to shift our focus toward label noise instead of altering features, and for adversarial we could try different attacks, namely *poisoning attacks*. As for the perturbation size norm, we could implement custom norms for tabular data, being able to simulate more closely real-world applications of our framework.

Regarding the robustness-enhancing mechanisms, we could shift our focus from adversarial robustness to robustness against other kinds of perturbations, such as data issues.

REFERENCES

[1] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 34, no. 11, pp. 2767–2787, 2010.

[2] A. Rajkomar, J. Dean, and I. Kohane, "Machine learning in medicine," *New England Journal of Medicine*, vol. 380, no. 14.

[3] T. Brennan, W. Dieterich, and B. Ehret, "Evaluating the predictive validity of the compas risk and needs assessment system," *Criminal Justice and behavior*, vol. 36, no. 1, pp. 21–40, 2009.

[4] G. Widmer and M. Kubat, "Effective learning in dynamic environments by explicit context tracking," in *Machine Learning: ECML-93: European Conference on Machine Learning Vienna, Austria, April 5–7, 1993 Proceedings 6*, pp. 227–243, Springer, 1993.

[5] L. Yang and A. Shami, "Iot data analytics in dynamic environments: From an automated machine learning perspective," *Engineering Applications of Artificial Intelligence*, vol. 116, p. 105366, 2022.

[6] A. Subbaswamy, R. Adams, and S. Saria, "Evaluating model robustness and stability to dataset shift," in *International conference on artificial intelligence and statistics*, pp. 2611–2619, PMLR, 2021.

[7] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *2017 ieee symposium on security and privacy (sp)*, pp. 39–57, Ieee, 2017.

[8] V. Ballet, X. Renard, J. Aigrain, T. Laugel, P. Frossard, and M. Detyniecki, "Imperceptible adversarial attacks on tabular data," *arXiv preprint arXiv:1911.03274*, 2019.

[9] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence–SBIA 2004: 17th Brazilian Symposium on Artificial Intelligence, Sao Luis, Maranhao, Brazil, September 29-Ocotber 1, 2004. Proceedings 17*, pp. 286–295, Springer, 2004.

[10] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[11] B. Frénay, A. Kabán, *et al.*, "A comprehensive introduction to label noise.," in *ESANN*, Citeseer, 2014.

[12] Y. Wu, E. Dobriban, and S. Davidson, "Deltagrad: Rapid retraining of machine learning models," in *International Conference on Machine Learning*, pp. 10355–10366, PMLR, 2020.

[13] S. Barocas, M. Hardt, and A. Narayanan, "Fairness in machine learning," *Nips tutorial*, vol. 1, p. 2017, 2017.

[14] M. Hardt, E. Price, and N. Srebro, "Equality of opportunity in supervised learning," *Advances in neural information processing systems*, vol. 29, 2016.

[15] A. Chouldechova, "Fair prediction with disparate impact: A study of bias in recidivism prediction instruments," *Big data*, vol. 5, no. 2, pp. 153–163, 2017.

[16] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq, "Algorithmic decision making and the cost of fairness," in *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*, pp. 797–806, 2017.

[17] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–35, 2021.

[18] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," *IEEE Transactions on Software Engineering*, 2020.

[19] U. Shaham, Y. Yamada, and S. Negahban, "Understanding adversarial training: Increasing local stability of supervised models through robust optimization," *Neurocomputing*, vol. 307, pp. 195–204, 2018.

[20] R. Mangal, A. V. Nori, and A. Orso, "Robustness of neural networks: A probabilistic and practical approach," in *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pp. 93–96, IEEE, 2019.

[21] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of statistical planning and inference*, vol. 90, no. 2, pp. 227–244, 2000.

[22] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate shift adaptation by importance weighted cross validation.," *Journal of Machine Learning Research*, vol. 8, no. 5, 2007.

[23] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, "Covariate shift by kernel mean matching," *Dataset shift in machine learning*, vol. 3, no. 4, p. 5, 2009.

[24] G. I. Webb, L. K. Lee, B. Goethals, and F. Petitjean, "Analyzing concept drift and shift from sample data," *Data Mining and Knowledge Discovery*, vol. 32, no. 5, pp. 1179–1199, 2018.

[25] P. Vorburger and A. Bernstein, "Entropy-based concept shift detection," in *Sixth International Conference on Data Mining (ICDM'06)*, pp. 1113–1118, IEEE, 2006.

[26] J. Perdomo, T. Zrnic, C. Mendler-Dünner, and M. Hardt, "Performative prediction," in *International Conference on Machine Learning*, pp. 7599–7609, PMLR, 2020.

[27] M. Hardt, N. Megiddo, C. Papadimitriou, and M. Wootters, "Strategic classification," in *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pp. 111–122, 2016.

[28] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," *Advances in neural information processing systems*, vol. 26, 2013.

[29] A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay, "Adversarial attacks and defences: A survey," *arXiv preprint arXiv:1810.00069*, 2018.

[30] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

[31] W. Brendel, J. Rauber, and M. Bethge, "Decision-based adversarial attacks: Reliable attacks against black-box machine learning models," *arXiv preprint arXiv:1712.04248*, 2017.

[32] J. Chen, M. I. Jordan, and M. J. Wainwright, "Hopskipjumpattack: A query-efficient decision-based attack," in *2020 ieee symposium on security and privacy (sp)*, pp. 1277–1294, IEEE, 2020.

[33] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[34] H. Chen, H. Zhang, D. Boning, and C.-J. Hsieh, "Robust decision trees against adversarial examples," in *International Conference on Machine Learning*, pp. 1122–1131, PMLR, 2019.

[35] A. Rezaei, A. Liu, O. Memarrast, and B. D. Ziebart, "Robust fairness under covariate shift," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 9419–9427, 2021.

[36] A. Mishler and N. Dalmasso, "Fair when trained, unfair when deployed: Observable fairness measures are unstable in performative prediction settings," *arXiv preprint arXiv:2202.05049*, 2022.

[37] J. Pombal, P. Saleiro, M. A. Figueiredo, and P. Bizarro, "Prisoners of their own devices: How models induce data bias in performative prediction," *arXiv preprint arXiv:2206.13183*, 2022.

[38] S. Milli, J. Miller, A. D. Dragan, and M. Hardt, "The social cost of strategic classification," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 230–239, 2019.

[39] H. Xu, X. Liu, Y. Li, A. Jain, and J. Tang, "To be robust or to be fair: Towards fairness in adversarial training," in *International Conference on Machine Learning*, pp. 11492–11501, PMLR, 2021.

[40] V. Nanda, S. Dooley, S. Singla, S. Feizi, and J. P. Dickerson, "Fairness through robustness: Investigating robustness disparity in deep learning," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 466–477, 2021.

[41] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, and G. Kasneci, "Deep neural networks and tabular data: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[42] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, "Learning under concept drift: A review," *IEEE transactions on knowledge and data engineering*, vol. 31, no. 12, pp. 2346–2363, 2018.

[43] D. Solans, B. Biggio, and C. Castillo, "Poisoning attacks on algorithmic fairness," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 162–177, Springer, 2020.

[44] F. Cartella, O. Anunciacao, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht, "Adversarial attacks for tabular data: Application to fraud detection and imbalanced data," *arXiv preprint arXiv:2101.08030*, 2021.

[45] S. Jesus, J. Pombal, D. Alves, A. Cruz, P. Saleiro, R. P. Ribeiro, J. Gama, and P. Bizarro, "Turning the tables: Biased, imbalanced, dynamic tabular datasets for ml evaluation," *arXiv preprint arXiv:2211.13358*, 2022.

[46] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.