# Project Work AI in Industry

**Nicola Palli**
Master's Degree in Artificial Intelligence, University of Bologna
nicola.palli@studio.unibo.it

## Abstract

In recent years, the pervasive integration of artificial intelligence (AI) systems within critical domains has elevated the importance of explainability within these fields. However, amidst these advancements, apprehensions have surfaced regarding the robustness and reliability of AI models, particularly in light of adversarial attacks. Adversarial AI [1] [2] denotes the phenomenon wherein maliciously crafted inputs are employed to deceive or manipulate AI systems, potentially resulting in erroneous outputs or security breaches. With the escalating adoption of AI across safety-critical applications such as healthcare, finance, and autonomous vehicles, comprehending the vulnerabilities posed by adversarial attacks is imperative. This study delves into various adversarial techniques, elucidating their mechanisms and demonstrating their efficacy in subverting classical explainability algorithms.

## 1 Introduction

Explainable artificial intelligence (XAI) methods, including post-hoc techniques like Partial Dependence Plots (PDP), Shapley Additive Explanations (SHAP), Local Interpretable Model-agnostic Explanations (LIME), Integrated Gradients (IG), and others, offer diverse mechanisms for interpreting the predictions of machine learning models. While XAI has found success in various applications such as autonomous driving and drug discovery, it faces criticism for its inability to faithfully explain complex black-box predictive functions. Nevertheless, recent studies on adversarial machine learning (AdvML) have shed light on the vulnerabilities of explanation methods, raising concerns about their trustworthiness and security. Adversarial attacks, such as data poisoning, model manipulation, and backdoors, have emerged as prominent failure modes of XAI methods, prompting the development of defense mechanisms like focused data sampling and model regularization. Fig. 1 illustrates common failure modes stemming from data and model influences on explanation outputs. The primary aim of attackers is to manipulate model explanations by altering data or models to deceive recipients. This poses a significant threat in sensitive applications, such as explaining decision-making systems used in legal contexts. Notably, attacks on explanations vary depending on the machine learning task, model architecture, and class of XAI methods, highlighting the need for systematic understanding and mitigation strategies.
In this project work, we explore several techniques of model manipulation aimed at deceiving and influencing machine learning models. The reliability of feature importance provided by traditional explainability techniques may not always be guaranteed: an apparent degree of fairness could potentially mask biases within the features, rendering them untrustworthy.

## 2 System description

This research on Adversarial AI in machine learning models builds upon a prior Project on Explainable AI [3], where various explainability algorithms like LIME, SHAP, and DiCE were evaluated. In the Explainability AI project, machine learning (ML) models were trained on established datasets tailored for predicting scores of robotic artistic performances using diverse input features. The employed model specifically is a Polynomial Regressor, with a focus on the results derived from the Linear Regressor, which is the Polynomial Regressor of degree 1.

The proposed model manipulation involves fine-tuning a pre-trained model with an objective function that integrates the standard classification loss with a penalty term based on the interpretation results. Accordingly, the overarching objective function for a model w, aimed at minimizing it for training data D utilizing the interpretation method

I, is formulated as follows:

$$L(D, w, w_0) = L_C(D, w) + \lambda L_F^I(D, w, w_0) \quad (1)$$

where $L_C$ is the original MSE loss function, $w_0$ are the original parameters of the model, $D$ is the dataset and $\lambda$ is a trade-off parameter.

- **Passive Fooling**: We define passive fooling as the act of inducing interpretation methods to produce uninformative explanations. To achieve this, *Location Fooling* is implemented. Here, we can directly choose which features should be more relevant for the model, by using the following penalty loss function:

  $$L_F^I(D, w, w_0) = \frac{1}{n} \sum_{i=1}^{n} ||h_i^I(w) - m||^2 \quad (2)$$

  where $n$ is the number of features, $h^I$ is the heatmap generated by a interpretation method $I$ for $w$ and $m \in R^n$ is a vector in which $m_i = 1$ if the $i_{th}$ feature must be relevant, and $m_i = 0$ otherwise

- **Active Fooling**: it's a concept where the goal is deliberately inducing interpretation methods to produce incorrect or misleading explanations by swapping the importance level between couples of features:

  $$L_F^I(D, w, w_0) = \frac{1}{n} \sum_{i,j} ||h_i^I(w_0) - h_j^I(w)||^2 \quad (3)$$

  where $n$ is the number of couples of features to swap and $h^I$ is the heatmap generated by a interpretation method $I$ for $w$.

There methods are exposed on paper [4], used in image processing field.

A second adversarial techinque is tested, in order to verify its effectiveness on fooling the classical and most used explainability techniques, such as LIME and SHAP [5].In particular, throught an adversarial model which exploits the individual predictions of a given black box model by constructing local interpretable approximations.

To gain insights into the synthetic data points generated through perturbations, we conducted an experiment as follows: Initially, we perturbed input instances using the method utilized by LIME, as described in the previous section. The synthetic data points resulting from input perturbations exhibit a significantly different distribution compared to the instances in the original input data. This observation suggests that discerning whether a data point is a product of perturbation or not is not a challenging task. Consequently, methodologies heavily reliant on such perturbations, such as LIME, can be manipulated.

This insight forms the basis of our proposed approach. By effectively distinguishing between data points originating from the input distribution and instances generated through perturbation, an adversary can construct an adversarial classifier (referred to as scaffolding) that mimics the behavior of the original classifier, potentially displaying extreme discriminatory tendencies on the input data points. However, this adversarial classifier behaves arbitrarily differently, appearing unbiased and fair, on the perturbed instances. Consequently, it effectively deceives techniques like LIME or SHAP into producing innocuous explanations. Subsequently, we formalize this intuition and elucidate our framework for constructing adversarial classifiers capable of misleading explanation techniques.

Let's denote the biased classifier described earlier as $f$. The adversary aims to develop a framework wherein post hoc techniques can be deceived into perceiving $f$ as innocuous when generating explanations for this black box model.

Recall that the real-world data, upon which this classifier is likely to be applied, follows a distribution $X_{dist}$, and $X$ represents a set of N data points sampled from this distribution, which the adversary has access to. To deceive the post hoc techniques, the adversary could devise an adversarial classifier that demonstrates biased behavior on instances sampled from $X_{dist}$ while maintaining an unbiased stance on instances that do not originate from $X_{dist}$.

Since the feature importances provided by LIME and SHAP heavily rely on perturbed instances (which may typically be Out-Of-Distribution (OOD) samples, the resulting explanations will portray the classifier designed by the adversary as innocuous.

Assuming $\psi$ represents an unbiased classifier (for instance, making predictions based on innocuous features that are uncorrelated with sensitive attributes), the adversarial classifier $e$ takes the following form:

$$e(x) = \begin{cases} f(x), & \text{if } x \in X_{\text{dist}} \\ \psi(x), & \text{otherwise} \end{cases}$$

To construct a classifier, is_OOD, which detects whether a given data point is an out-of-distribution (OOD) sample (i.e., is_OOD$(x)$ = True) or not, we proceed as follows:

1. We create a new dataset from the instances in $X$ by perturbing each instance $x$ to generate a new instance $x_p$. The details of perturbation are provided in the Experimental Evaluation section.

2. Denote the set of all instances generated via perturbation as $X_p$.

3. The instances in $X$ are then assigned the class label `False`, indicating that they are not OOD samples. Conversely, the instances in $X_p$ are assigned the class label `True` unless they are already in $X$.

4. We proceed to train an off-the-shelf classifier on the combined dataset $X \cup X_p$ and their corresponding class labels.

This process enables the creation of a classifier capable of distinguishing between in-distribution and out-of-distribution samples based on the characteristics of the perturbed instances.

## 3 Data

Two distinct datasets were obtained from the University of Bologna [6], focusing on statistical information pertaining to robotic dance performances and audience reception. These datasets diverge based on evaluations conducted by audiences with contrasting backgrounds—one with an artistic orientation and the other with a scientific background. They encompass a diverse array of features, including categorical aspects such as music genre and numerical values like beats per minute (bpm). Additionally, both datasets include seven target variables reflecting evaluations of rhythm, storytelling, public involvement, and other parameters. Our models are tasked with generating predictions for each of the seven target variables across both datasets.

The preprocessing strategy involved transforming the initial input features to yield 27 distinct features. These features encompassed the original numerical values, which were standardized by removing the mean and scaling to unit variance. Discrete categorical levels were derived from the categorical values. Notably, the "AI_technique" categorical input feature was divided into three boolean variables—AI_planning, AI_searchstrategy, and AI_constraint—representing different AI techniques. Similarly, the musicGenre feature was divided into mPop, mRock, mElectric, mFolk, and mIndie categories.

This transformation process was facilitated by two dedicated pipelines—one for numerical features and another for categorical features. For numerical features, our pipeline executed a two-phase transformation. Firstly, it employed a median imputation strategy to address missing values within numerical features, replacing them with the corresponding median values to ensure robust data handling. Following imputation, the pipeline scaled the numerical features using an appropriate scaler. On the categorical side, the process began with the separation of distinct values for each categorical feature, followed by one-hot encoding of each isolated categorical value.

Upon applying this preprocessing step, our final dataset comprised 27 variables, including the original 18 numerical features supplemented by an additional 9 features capturing the diverse values derived from the categorical ones.

## 4 Experimental setup and results

In this section, we delve into the experimental setup that forms the foundation of our findings. Following this, we unveil the outcomes derived from a comprehensive exploration, where we deploy various model manipulation techniques and leverage existing explainability algorithms. Through the presentation of these results, we aim to shed light on the boundaries and vulnerabilities inherent in explainable AI.

Our evaluation process involves comparing the feature importance assigned by unaffected models with that of manipulated ones. Rather than relying solely on traditional metrics, our focus is on gauging how effectively models attribute importance values to pertinent features. Specifically, the Linear Regressor coefficients, LIME, SHAP, and DiCE are used as indicative of fair feature importance. These established benchmarks serve as a baseline for evaluating the efficacy of the adversarial methodologies emploied.

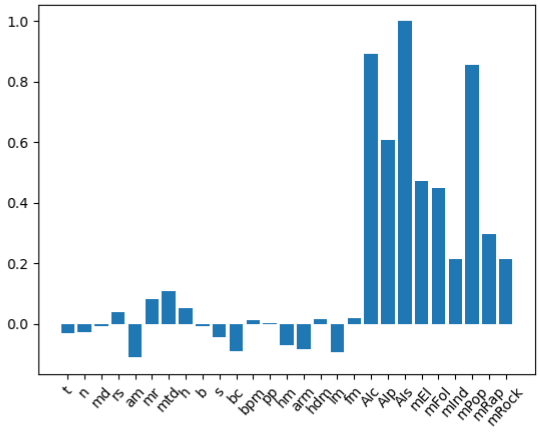

Figure 1: Feature importance of the fair model.

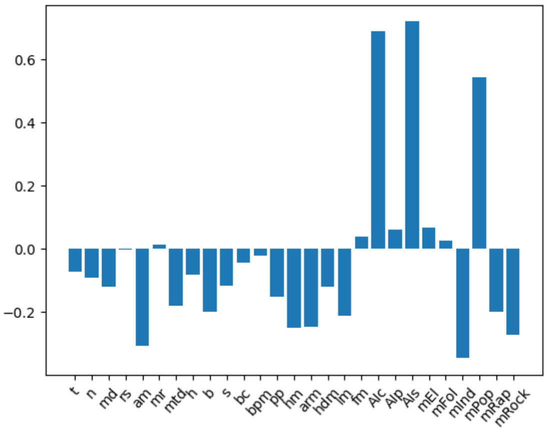

Figure 2: Feature importance after Location Fooling

In this report, we focus solely on presenting the outcomes of the *Evaluation on Human Reproducibility* conducted using the *Scientific Dataset*. However, it's worth noting that all the findings discussed herein are publicly available on our GitHub repository [7]. Notably, our experiments include Location Fooling, where we attempted to exclusively emphasize the time feature 't', and Active Fooling, where we exchanged the relevance of 'mPop' and 'rs'—one being among the most significant features and the other among the least relevant.

| Results | Fair lr | Location Fool | Active Fool |
|---|---|---|---|
| MAE | 0.93 | 1.06 | 0.94 |
| MSE | 1.38 | 1.75 | 1.39 |
| RMSE | 1.17 | 1.32 | 1.18 |
| MPE | 13.21 | 15.14 | 13.35 |

Table 1: Performences of the three regressors on test set.

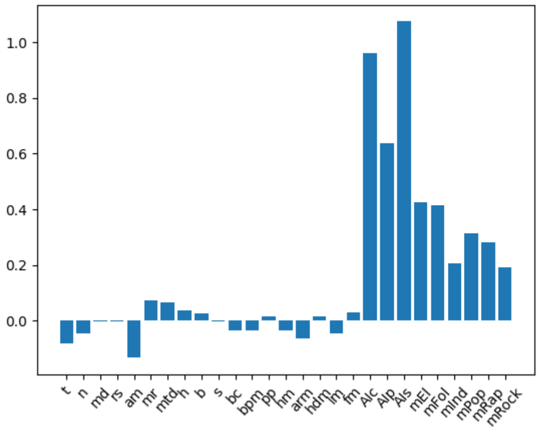

Figure 3: Feature importance after Active Fooling

The results indicate comparable performances among the three models, albeit with a significant disparity in feature relevance. Location Fooling resulted in a decrease in the importance of the most relevant features, notably increasing the importance of 't' from 0.029 to 0.072. Regarding Active Fooling, the importance of unchanged features remained relatively consistent, while 'mPop' saw a substantial reduction from 0.856 to 0.313. Further increases in feature importance changes can be achieved by adjusting the parameter $\lambda$, albeit with a potential trade-off in performance. Notably, all results were obtained with $\lambda = 10$.

Finally, we test the adversarial model $e(x)$. For each sample in the dataset, three new counterfactuals are generated using DiCE. These newly generated samples are labeled as $OOD = True$, while the original samples are labeled as $OOD = False$. Subsequently, a Random Forest classifier consisting of 100 Decision Trees is trained to distinguish between Out-of-Distribution (OOD) samples and in-distribution samples. Additionally, two linear regressors, $f(x)$ and $\psi(x)$, are trained on original and OOD samples, respectively.

Specifically, $\psi(x)$ is trained by incorporating the penalty loss function:

$$L_F^I(D,w) = \frac{1}{n_c}\sum_{i\in C} ||h_i^I(w) - 1||^2 \qquad (4)$$

In particular, $\psi(x)$ is trained with the objective of emphasizing the importance of the time feature 't', although it can be trained to consider various other features as well. The set $C$ represents the features that are mandated to be relevant.

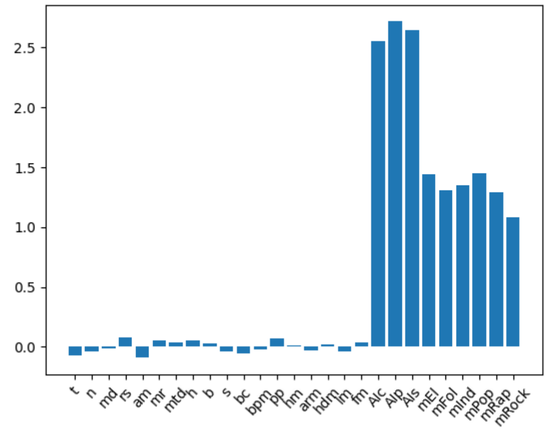

Figure 4: Feature importance of f(x)

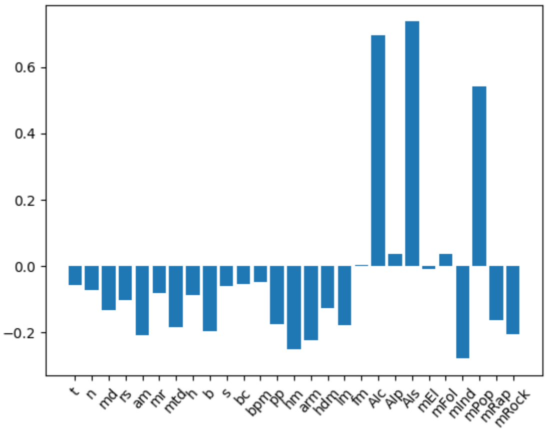

Figure 5: Feature importance of $\psi(x)$

The LIME results in Figure 6 indicate that the feature importance computed by this algorithm aligns with the feature importance of $f(x)$, even when the original data are evaluated by $\psi(x)$. Notably, the most relevant features identified by LIME (*AItechnique planning*, *AItechnique searchStrategy*, *AItechnique constraints*, *musicGenre Electronic*, and *musicGenre Pop*) correspond closely to the most relevant features for $f(x)$. In contrast, $\psi(x)$ yields markedly different results, as *AItechnique planning*, *AItechnique constraints*, and *musicGenre Electronic* exhibit minimal relevance.

## 5 Conclusion

In this section we will provide a discussion on the results presented in the previous section.

In conclusion, our study offers a comprehensive exploration into the nuances of explainable AI (XAI) methodologies and their susceptibility to manipulation and adversarial attacks. By delving into the experimental setup and unveiling the outcomes derived from our investigations, we have shed light on both the capabilities and limitations of existing XAI techniques. Findings highlight the intricate interplay between model manipulation, feature relevance, and model performance, underscoring the importance of robust and reliable interpretability in AI systems. The observed disparity in feature importance between unaffected models and manipulated ones underscores the need for critical evaluation and validation of XAI outputs, also when employing classical algorithms such as LIME and SHAP. Furthermore, the evaluation of the adversarial model emphasizes the necessity of developing robust defenses against potential attacks aimed at undermining the interpretability and trustworthiness of AI models. Therefore, future research efforts should focus on refining and enhancing XAI techniques to bolster their resilience and efficacy in real-world applications, thereby fostering greater trust and understanding between AI systems and their human users.

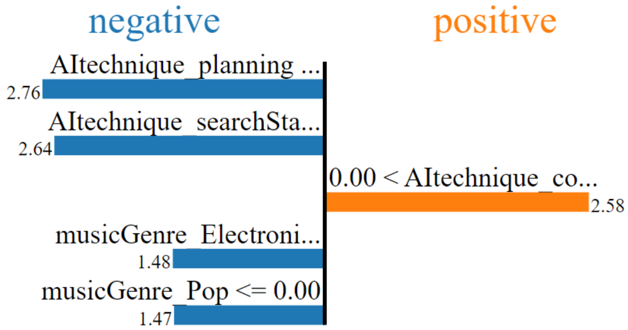

Figure 6: Feature importance using LIME

## References

[1] Hubert Baniecki and Przemyslaw Biecek. "Adversarial attacks and defenses in explainable artificial intelligence: A survey". In: *Information Fusion* (2024), p. 102303.

[2] Reza Shokri et al. "Membership inference attacks against machine learning models". In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18.

[3] Marvasi Riccardo Palli Nicola Saturno Edoardo. *AIinIndustryRoboticDancePerformanceEvaluation*. https://github.com/NicolaP00/AI_in_Industry-Robotic_Dance_Performance_

Evaluation.git. Accessed: March 2024. 2024.

[4] Juyeon Heo, Sunghwan Joo, and Taesup Moon. “Fooling neural network interpretations via adversarial model manipulation”. In: *Advances in neural information processing systems* 32 (2019).

[5] Dylan Slack et al. “Fooling lime and shap: Adversarial attacks on post hoc explanation methods”. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 2020, pp. 180–186.

[6] *RoboticDancePerformanceEvaluation*. https://github.com/ProjectsAI/RoboticPerformanceArtisticEvaluation.git. Accessed: March 2024. 2022.

[7] Palli Nicola. *PWIndustry*. https://github.com/NicolaP00/PW_Industry.git. Accessed: March 2024. 2024.