Measuring Fairness in Machine Learning models via Counterfactual Examples

Rami Haffar, Ashneet Khandpur Singh, Josep Domingo-Ferrer, and Najeeb Jebreel

Universitat Rovira i Virgili,
Department of Computer Engineering and Mathematics,
CYBERCAT-Center for Cybersecurity Research of Catalonia,
Av. Països Catalans 26, 43007 Tarragona, Catalonia
{rami.haffar, ashneet.singh, josep.domingo, najeeb.jebreel}@urv.cat

Abstract. Machine learning has become a vital resource of the modern society. It is present in everything around us, from a smartwatch to a self-driving car. To train a machine learning model, a heap of data is used. This can be worrisome in the case of that learned models can be discriminatory with respect to protected features such as race or gender. In order to develop fair models and verify the fairness of these models, a plethora of work has emerged in recent years. In this work, we propose a method, based on counterfactual examples, that detects any bias in the machine learning model. Our method works for different data types, including tabular data and images.

Keywords: Machine Learning · Fairness · Counterfactual Examples · Adversarial Examples · GANs.

1 Introduction

Artificial intelligence (AI) is undergoing a rapid evolution. Most companies that were evaluating or experimenting with AI not so long ago are now using it. Machine learning (ML) has played a vital role in this advancement. Various industries ranging from information technology, finance, media, gaming, robotics, have already set ML technology in practice.

However, machine learning is still in its early stages and plenty of work needs to be done. In the future, ML will help build self-learning robots and machines that are expected to improve their performance without using any human involvement. In this way, machines will be able to make decisions based on past data to predict the best future action.

Since ML models use high amounts of data, concerns have arisen that learned models may be discriminatory with respect to sensitive features, e.g. race, gender, and socioeconomic status. As a result of these issues, an astounding number of methods for developing fair models and verifying the fairness of existing models have emerged in recent years.

There are two types of fairness definitions that have been considered in the literature [7], [15]:

- The vast majority of definitions deal with group fairness, e.g. demographic parity [5], equalized odds [10] or predictive parity [3]. Basically, such definitions pre-identify what attributes or groups should be protected. Group fairness metrics have the advantage that they can be easily computed and they reflect anti-discrimination legislation. These metrics have a solid theoretical foundation as well as practical methods and implementations. Their weakness is that they entail no guarantees for individuals.
- The second type of definitions are about individual fairness. They assume a similarity metric of the individuals for the classification task at hand that is generally hard to find. For instance, unawareness [5] and counterfactual fairness [12]. This type of definitions bind at the individual level.

Any of the metrics mentioned above can be used to calculate disparities in data across groups, but many of them cannot be balanced across subgroups at the same time. As a result, one of the most crucial components of measuring bias in the model is understanding how fairness should be defined for a certain scenario. In this work, we are going to merge the concept of group fairness with that of individual fairness, in the sense that the proposed method protects the individuals of any minority against any bias in the model.

Contribution and plan of this paper

The purpose of this paper is to detect any bias in the ML models targeting any individual, regardless of the type of data used in the model. To that end, we leverage counterfactual examples. The key contributions of the paper are: (i) to detect the bias with regard to the ML model behavior and the training data, unlike the previous work [12] that solely focuses on the bias of the training data; (ii) to provide bias detection regardless of the data type (in particular for tabular data and images), thanks to the use of counterfactual examples, which offer developers a straightforward way to measure the fairness of their models.

The rest of this paper is organized as follows. Section 2 introduces the three different fields of ML used in this work and the related work. Section 3 presents the proposed methodologies to measure fairness in ML models. Section 4 reports experimental results. Finally, Section 5 gathers conclusions and sketches future research lines.

2 Related Work

This section gives an overview of three distinct fields of machine learning research that are explicitly merged in this paper.

2.1 Fairness

The existing research addressing the topic of fairness in machine learning has focused on how to measure and evaluate fairness (or, equivalently, bias) in models.

The work in [5] introduced the definition of individual fairness, which contrasts with group-based notions of fairness [10, 3] that require demographic groups to be treated similarly on average.

Later, [12] introduced a causal, individual-level, definition of fairness, called counterfactual fairness, which states that a decision is fair toward an individual if it coincides with the one that would have been taken in a counterfactual world in which the sensitive attribute was different. More formally, given protected attributes A, remaining attributes Z and output of interest Y, a classifier $\hat{Y}$ is counterfactually fair if, under any context $A = a$ and $Z = z$, we have

$$P(\hat{Y} = \hat{y}_{A=a} | A = a, Z = z) = P(\hat{Y} = \hat{y}_{A=a'} | A = a, Z = z),$$

for any value a' attainable by A.

In relation with group fairness, counterfactual fairness is complementary to the group fairness notion of equality of odds, which demands equality of true positive rates and true negative rates for different values of the sensitive attribute.

2.2 Adversarial examples

Adversarial examples are inputs to machine learning models that an attacker intentionally constructs to fool the model into returning a false prediction [14]. Adversarial examples are closely related to counterfactual examples: [16] characterize counterfactuals as adversarial examples that perturb inputs in human-interpretable and possibly problematic ways. Thus, whereas adversarial examples try to deceive the model, counterfactual examples try to understand or explain it.

Formally speaking, given a input x, a classifier $\hat{Y}$, a distance metric d (which can be the L_p distance) and a small distance $\epsilon > 0$, an adversarial example is defined as an input $x' \neq x$, such that $d(x', x) < \epsilon$ and $\hat{Y}(x') \neq \hat{Y}(x)$.

In [18], by utilizing adversarial examples for data augmentation, the authors implemented a prototype application to solve the algorithm bias problem. In order to obtain a fair dataset in which the distribution of bias variables is balanced, they apply adversarial attacks to generate examples containing information of bias variables as the enhanced data.

2.3 GANs

Generative Adversarial Networks, or GANs for short, are a type of generative model, that is, a model that can produce new content based on its training data. They were first introduced by [8] and they have a variety of applications. However, their most common use is to generate new images. A GAN consists of two artificial neural networks, a generator G and a discriminator D, that compete against each other. G creates new data instances while D evaluates them for authenticity.

More specifically, G takes as input a random noise vector z and outputs an image $z = G(z)$. D receives as input either a training image x or a synthesized image $G(z)$ from G and outputs a probability distribution $D(\cdot)$ over possible image sources. D is trained to maximize the log-likelihood it assigns to the correct source:

$$L = E_x[\log(D(x))] + E_z[\log(1 - D(G(z)))].$$

The generator G is trained to minimize $E_z[\log(1-D(G(z)))]$, that is, to minimize the likelihood that its generated images are detected as fake by D.

In [17], the authors develop a GAN for fairness. Their architecture is limited to low-dimensional structured data and only applies to demographic parity. In contrast, our work is geared towards high-dimensional image data and individual fairness.

Another work that measures fairness based on GANs is [1]. The authors propose an adversarial approach, inspired from GANs, in which a sanitizer is learned from data representing the population. In this work, local sanitization is employed to reach algorithmic fairness.

3 Measuring Fairness via Counterfactual Examples

In this section, we introduce our proposed method to measure fairness in ML models. Our approach rests on generating counterfactual examples. To this end, the technique used to generate those counterfactuals may differ depending on the input data type. We next discuss the cases of tabular data and image data.

3.1 Measuring fairness in tabular data

We want to measure the fairness of any ML model, which will ensure that the predictions of the model are not biased against any minority in the data set. To this end, we need to guarantee that the ML model is not making its predictions based on a sensitive attribute. The fairness assurance process should be automated in the sense that it is not enough to change the attribute value and monitor the outcome of the model. At the same time, this process should be model- and data-related, because the bias might be in the training data or in the learned model.

In order to measure the fairness of an ML model f, for a specific data record x, taking into consideration the protected attribute a, we have to satisfy the counterfactual fairness definition mentioned in Section 2.1. We propose an automated approach that creates counterfactual examples for each record in the data set. The proposed method uses adversarial examples as a means to create those counterfactual examples. To create an adversarial example, the proposed method targets only the attributes that have the highest effect on the ML model prediction, in an attempt to alter its predicted label with the smallest input changes. In this situation, if the changed attribute is one of the protected attributes, this indicates that the model is not fair to this record.

Algorithm 1 describes the proposed method. Given an ML model f, a maximum allowed value ϵ_{max}, an input record x containing n attributes, the method generates the counterfactual example x_* by changing at most $c \leq n$ attributes from x, where c is the number of protected attributes. First, to set the target classification label y_* we desire for x_*, we compute *probs*, the probability vector that f outputs for x. Then, we set the desired class label y_* to be the index of the second most probable class in *probs*. Choosing this class makes it easier for the proposed method to find the desired x_* with small changes to x. Note that the user can also set y_* to any class label she wants. Then, to select the c attributes which have the highest effect on the model prediction, we compute the gradient of the loss between the model output $f_u(x)$ and the desired output y_* with regard to the attributes of the input record. After that, we take the L_1-norm of the computed gradient ∇_x. Subsequently, we identify the c attributes with the highest L_1-norms as the attributes to be changed when generating x_*. This is done using a weighting vector w which contains 0s for the unchanged attributes and 1s for the changed attributes. Using this vector allows us to change only c attributes while creating x_*. Afterwards, we keep repeating the gradient descent step in expression

$$x_* = x - w \cdot \epsilon \cdot \frac{\partial}{\partial x} \mathcal{L}(f(x), y_*),$$

with regard to w until one of two following conditions is satisfied: i) an adversarial example x_* is obtained that fools f into labeling it as y_* or ii) the maximum value ϵ_{max} is reached for ϵ (which means that no x_* close to x could be found). Note that we start with a small $\epsilon = 0.05$ and we increase it by 0.05 at each step. Once we create the adversarial example x_*, we compare it with x to identify the changed attributes. If one or more of the changed attributes belong to a, the set of the protected features, then this indicates bias in the model.

3.2 Measuring fairness in image data

Measuring fairness for image classifiers is more challenging than measuring fairness for tabular data classifiers because the attributes in images are not self-explanatory. To tackle this challenge, we propose a method that generates counterfactual examples for the image data by leveraging the GANs described in Section 2.3.

In the proposed method, we use two generators and one discriminator consisting of a trained binary classifier, as shown in Fig.1. Both generators create images to fool the trained image classifier model. However, each generator can only fool the model into one specific outcome. Generator A creates counterfactual images to be classified by the classifier into class a, whereas generator B's counterfactuals are to be classified into class b. After training both generators, they can be used to create counterfactual examples to detect any bias in the classifier. Any image m from class a that the classifier misclassifies into class b is passed through generator A to generate the counterfactual example m_*. This counterfactual m_* will be classified into class a. By comparing m and m_* we can

Algorithm 1 Creating the counterfactual example

Input: Trained model f, record x, maximum allowed ϵ_{max}, maximum number of attributes to be changed c.
Output: Counterfactual example x_*
$probs \leftarrow$Get_Probabilities($f(x)$)
$y_* \leftarrow argmax(probs, 2)$
$n \leftarrow$ Number of attributes in x
$|\nabla_x| \leftarrow abs(\frac{\partial}{\partial x}\mathcal{L}(f(x), y_*))$
$w \leftarrow$ Zero vector of length n
$idxs \leftarrow$ Indices of the highest c values in $|\nabla_x|$
for $idx \in idxs$ **do**
    $w[idx] = 1$
end for
$x_* \leftarrow x$
$\epsilon \leftarrow 0.05$
while $f(x_*) \neq y_*$ and $\epsilon \leq \epsilon_{max}$ **do**
    $x_* \leftarrow x - w \cdot \epsilon \cdot \frac{\partial}{\partial x}\mathcal{L}(f(x), y_*)$
    $\epsilon \leftarrow \epsilon + 0.05$
end while
if $f(x_*) = y_*$ **then**
    Return x_*
else
    Return $\emptyset$
end if

notice whether they differ in any of the discriminatory attributes (those related to race, such as dark skin or dark hair). Similarly, an image m' from class b misclassified into class a is passed through generator B to create the counterfactual m'_*.

4 Empirical Results

In this section, we evaluate the performance of the proposed approach on two ML tasks: tabular data classification and image classification. For each task, we trained a baseline model with the original data set and a biased model after we did some alterations to the data set. In both data sets, the baseline and biased models had the same architecture. First, we show the performance of the models and then we evaluate the proposed fairness measures.

4.1 Experimental setup

Data sets and provider models. We evaluated the proposed approach on two data sets:

– **Adult**[1] is a data set from the UCI Machine Learning repository[4]. It is a tabular data set that contains $48,842$ records of census income information

[1] https://archive.ics.uci.edu/ml/datasets/adult

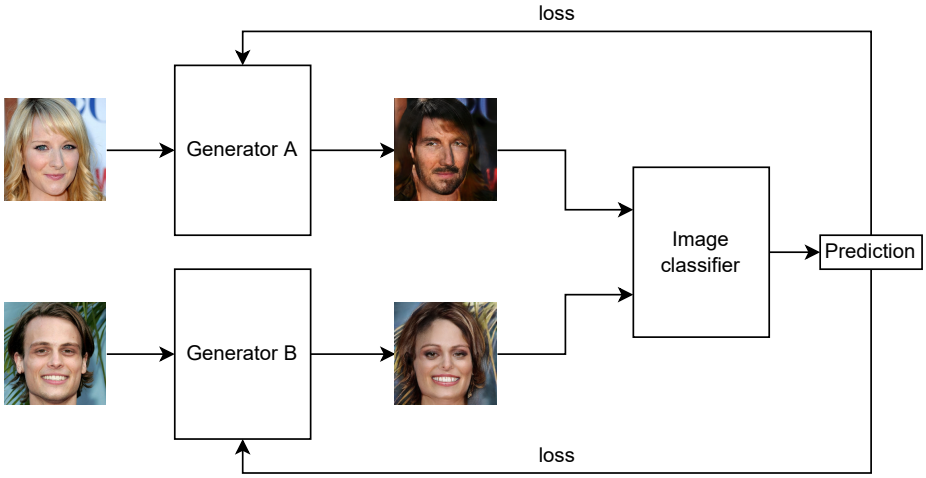


Fig. 1: Training the generator of the counterfactual examples.

with 14 numerical and categorical attributes. From these, we dropped the final weights (*fnlwgt, capital-loss, and capital-gain*), which reveal too much information to the model, and the *education* attribute, which is redundant with *education-num*. We also encoded categorical attributes as numbers. The class label is the attribute *income*, that classifies records into either > 50K or ≤ 50K. We used 80% of the data as training data, and the remaining 20% as validation data.

– **CelebA**[2] is a binary classification data set from Kaggle [13]. It consists of RGB images of male and female faces with a training set of 17,943 female images and 10,057 male images. The validation set contains 2,000 images evenly divided into the two classes. Since the images have large sizes of 1024 $\times$ 1024 pixels, we first resized them to 256$\times$256 pixels in order to train our models faster.

The architectures of the models used in the experiments are shown in Table 1. We took the same models from [9].

In all experiments, we employed the binary cross-entropy loss function and the Adam optimizer [11].

Training data for biased models. The two original data sets were balanced in terms of sensitive data. To train biased models, we modified both training data sets in order to train the biased model.

– **Adult:** To drive the model to be biased against the attribute *gender*, we selected 45% of the *females* whose *race* was *black* and whose *income* was >50K, and we changed their *income* to ≤ 50K. Also, we selected 45% of the

[2] https://www.kaggle.com/dataset/504743cb487a5aed565ce14238c6343b7d650ffd28c071f03f2fd9b25819e6c9

Table 1: Architectures of the models used in the experiments for the Adult and CelebA classification data sets. $C(3, 32, 3, 0, 1)$ denotes a convolutional layer with 3 input channels, 32 output channels, a kernel of size 3×3, a stride of 0, and a padding of 1; $MP(2, 2)$ denotes a max-pooling layer with a kernel of size 2×2 and a stride of 2; and $FC(18432, 2048)$ indicates a fully connected layer with 18,432 inputs and 2,048 output neurons. We used $ReLU$ as an activation function in the hidden layers; lr stands for learning rate.

Data set	Model architecture	Hyper-parameter
Adult	FC(12,100), FC(100,100), FC(100,2)	lr = 0.001, epochs = 10, batch = 128
CelebA	C(3,32,3,0,1), C(32,64,3,1,1), MP(2,2),C(64,128,3,0,1), C(128,256,3,1,1), MP(2,2), C(256,512,3,0,1), C(512,512,3,1,1), MP(2,2), C(512,256,3,0,1), C(256,256,3,1,1), MP(2,2), FC(16384,2048), FC(2048,1024), FC(1024,512), FC(512,128), FC(128,32), FC(32,2)	lr = 0.001 epochs = 10 batch = 64

males whose *race* was *black* and whose income was $\leq$50K, and we changed their income to >50k.

– **CelebA:** To train a biased gender classification model, we separated from the female training data the images containing dark skin. In order to do so, we used the unsupervised cluster K-means [2], while keeping track of some cherry-picked images to monitor the outcome of the cluster. The desired class consisted of 1904 images. We changed the class of 60% of those images from female into male.

Evaluation metrics. We used the following evaluation metrics to measure the performance of the trained surrogate models and the generated explanations:

– *Accuracy*: Number of correct predictions divided by the total number of predictions. We used this metric to measure and compare the performance of the baseline and the biased models.
– *ROC curve[6]*: The Receiver operating characteristic (ROC) is a graph plotting the false positive rate in the abscissae and the true positive rate in the ordinates. It shows the performance of a classification model at all classification thresholds. The AUC (area under the ROC curve) takes values from 0.0 (when the model predictions are 100% wrong), to 1.0 (when the model predictions are 100% correct). AUC is scale invariant, and classification-threshold invariant. In some cases it is a better metric than accuracy.

In this work, we evaluated the fairness of the model based on its performance on the targeted minority, since there are no methods in the literature to compute the individual fairness score.

4.2 Results

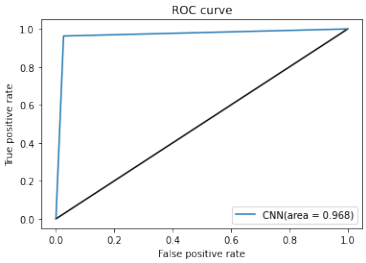
Accuracy and ROC-AUC score. In Table 2, we show the accuracy of the baseline and biased models when evaluated on the full evaluation data set, for

both data sets. The baseline model was more accurate than the biased model on both data sets, even though the biased model's accuracy was not much lower than the baseline model's.

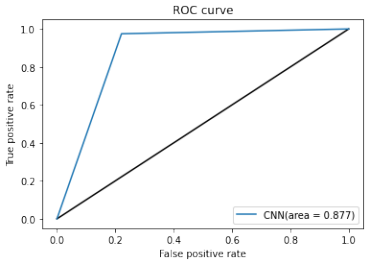
Table 2: Accuracy of the baseline and biased model evaluated on the full evaluation data set, for the Adult and the CelebA data sets

Data set	Baseline model	Biased model
Adult	82.80%	79.67%
CelebA	98.52%	86.42%

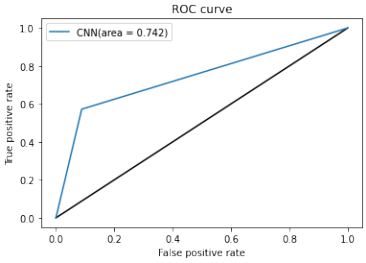
In Fig. 2, ROC curves are presented for both models on the two data sets. The results are consistent with those of Table 2, where the baseline and the biased models exhibit small performance differences.



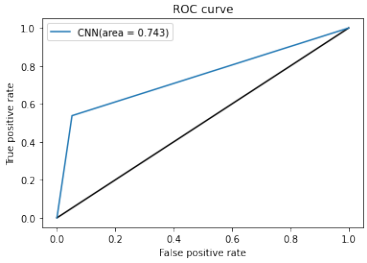
(a) ROC curve of the baseline model on the Adult data set



(b) ROC curve of the biased model on the Adult data set



(c) ROC curve of the baseline model on the CelebA data set



(d) ROC curve of the biased model on the CelebA data set

Fig. 2: ROC curve of the baseline and biased models on both data sets

To better understand the performance differences between the baseline and the biased models, we evaluated them on modified data sets, which consist of the samples belonging to the minorities in each data set. On the one side, we

derived an evaluation set from Adult by picking all the records with *race* equal to *black*. On the other hand, we derived an evaluation set from CelebA by picking all female images with dark skin. We call those evaluation sets the targeted data sets. Table 3 reports the accuracy of the models when they were evaluated on the targeted sets. For both data sets, the baseline model accuracy was similar to that in Table 2, while the performance of the biased model was very poor: it was around 1% in both data sets. This illustrates that the biased models were precisely biased against the individuals in the targeted data sets.

Table 3: Accuracy of the baseline and biased models evaluated on the targeted data sets derived from Adult and CelebA

Data set	Baseline model	Biased model
Adult	81.90%	0.54%
CelebA	99.60%	1.67%

Fairness of the trained models.

– **Adult data set:** We generated a counterfactual example for each record in the validation portion of the Adult data set. First, we considered the attribute *gender* as the protected attribute. Then, we used Algorithm 1 with $c = 1$, to restrict the changes to only one attribute when generating the examples. After that, we computed the number of cases where the counterfactual example was created by changing the protected attribute. We found that for the biased model, this happened 1,637 times, while for the baseline model this happened only 12 times. These results indicate that for the baseline model the protected attribute was not essential in the prediction. On the other hand, the protected attribute had a very high impact on the predictions made by the biased model. We present two examples of the counterfactual examples generated by the proposed method in Table 4. Record 1's income was classified as $> 50K$ by both models. The first counterfactual example which was created based on the baseline model shows that the original record's prediction can change to $\leq 50K$ by reducing the *educational level* from 13 to 9. This recommendation from the proposed method seems logical in the sense that less education yields decreased income. The second counterfactual example which was created based on the biased model recommended changing the *gender* from *male* to *female*, which shows that the model learned a gender bias (income of males is higher than the income of females). On the other hand, Record 2's income was classified as $\leq 50K$ by both models. In this case, the counterfactual created by the baseline recommended increasing the *hours-per-week* from 47 to 60, which is also reasonable because more working hours usually means more payout. In contrast, the counterfactual created by the biased model recommended to change the attribute *gender*

but this time from *female* to *male*, because the model learned that males earn more.

Table 4: Two examples of records from the Adult data set. Symbol '"' indicates that the value of the attribute did not change during the creation of the counterfactual example.

Features	age	workclass	educational -num	marital -status	occupation	relationship	race	gender	hours -per -week	native -country	income
Original record 1	38	State-gov	13	Married	Protective-serv	Husband	Black	Male	52	United-States	>50K
Baseline recommendation	"	"	9	"	"	"	"	"	"	"	$\leq 50K$
Biased model recommendation	"	"	"	"	"	"	"	Female	"	"	$\leq 50K$
Original record 2	43	Local-gov	14	Unmarried	Tech-support	Not-in-family	Black	Female	47	United-States	$\leq 50K$
Baseline recommendation	"	"	"	"	"	"	"	"	60	"	> 50K
Biased model recommendation	"	"	"	"	"	"	"	Male	"	"	> 50K

– **CelebA data set:** In order to generate the counterfactual examples for the CelebA data set, we selected the female images that were classified into the label male by the biased model and into the label female by the baseline model, because those images were targeted when training the biased model. Even though those images correspond to females, we pass them through the female generator, that is due to the classifier (in this case it is also the discriminator) to classify them as a male. Fig. 3 presents five examples of the counterfactual created by the proposed method. In all the images, the proposed method made the skin lighter, and the hair color blond. Those counterfactuals were classified as females by both models. Thus, our method detected that the biased model classified every face image with dark skin and black hair as male.



Fig. 3: Five examples of the counterfactual examples created by the proposed method

5 Conclusion and Future Work

In this paper, we have examined fairness in the scenario of binary classification for two types of input data, tabular and image data. The proposed method is based on generating counterfactual examples to measure fairness in ML models. In the case of tabular data, we used adversarial examples to create the counterfactuals. To achieve this for image data, we used GANs as a generator for the counterfactuals. Our experiments confirm that the proposed method can detect any model bias against protected attributes. As future work, we plan to implement a new measure that characterizes fairness for image classifier models, since there is a limitation in the current literature on this topic. Furthermore, we intend to automate the comparison process between the original data and the counterfactual examples. We also envision generalizing our method to non-binary classification.

Acknowledgements We acknowledge support from the European Commission (projects H2020-871042 "SoBigData++" and H2020-101006879 "MobiDataLab") and from the Government of Catalonia (ICREA Acadèmia Prize to J. Domingo-Ferrer).

References

1. Aïvodji, U., Bidet, F., Gambs, S., Ngueveu, R. C., & Tapp, A. (2021). Local data debiasing for fairness based on generative adversarial training. Algorithms, 14(3), 87.
2. Ball, G. H., & Hall, D. J. (1965). ISODATA: a novel method of data analysis and pattern classification. Stanford Research Inst, Menlo Park CA.
3. Chouldechova, A. (2017). Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. Big Data 5(2), 153-163.
4. Dua, D., & Graff, C. (2017). UCI Machine Learning Repository http://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science.
5. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., & Zemel, R. (2012). Fairness through awareness. In Proceedings of the 3rd innovations in theoretical computer science conference (pp. 214-226). ACM.
6. Fawcett, T. (2006). An introduction to ROC analysis. Pattern Recognition Letters, 27(8), 861-874.
7. Gajane, P., & Pechenizkiy, M. (2018). On formalizing fairness in prediction with machine learning. arXiv preprint arXiv:1710.03184.
8. Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014) Generative Adversarial Networks. In Advances of Neural Information Processing Systems-NIPS 2014, 27.
9. Haffar, R., Jebreel, N. M., Domingo-Ferrer, J., & Sánchez, D. (2021). Explaining image misclassification in deep learning via adversarial examples. In International Conference on Modeling Decisions for Artificial Intelligence-MDAI 2021 (pp. 323-334). Springer, Cham.

10. Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. In Advances in Neural Information Processing Systems-NIPS 2016, 29.
11. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
12. Kusner, M. J., Loftus, J., Russell, C., & Silva, R. (2017). Counterfactual fairness. In Advances in Neural Information Processing Systems-NIPS 2017, 30.
13. Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In Proceedings of the IEEE international conference on computer vision (pp. 3730-3738).
14. Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. & Fergus, R. (2014). Intriguing properties of neural networks. In International Conference on Learning Representations. arXiv preprint arXiv: 1312.6199.
15. Verma, S., & Rubin, J. (2018). Fairness definitions explained. In 2018 IEEE/ACM International Workshop on Software Fairness (FairWare 2018) (pp. 1-7). IEEE.
16. Wachter, S., Mittelstadt, B. D. & Russell, C. (2017). Counterfactual explanations without opening the black box: automated decisions and the GDPR. CoRR abs/1711.00399.
17. Xu, D., Yuan, S., Zhang, L. & Wu, X. Fairgan (2018). Fairness-aware generative adversarial networks. In 2018 IEEE International Conference on Big Data (Big Data), (pp. 570–575).
18. Zhang, Y., & Sang, J. (2020). Towards Accuracy-fairness paradox: adversarial example-based data augmentation for visual debiasing. In Proceedings of the 28th ACM International Conference on Multimedia (pp. 4346–4354).