DENOUNCER: Detection of Unfairness in Classifiers

Jinyang Li
University of Michigan
jinyli@umich.edu

Yuval Moskovitch
University of Michigan
yuvalm@umich.edu

H. V. Jagadish
University of Michigan
jag@umich.edu

ABSTRACT

The use of automated data-driven tools for decision-making has gained popularity in recent years. At the same time, the reported cases of algorithmic bias and discrimination increase as well, which in turn lead to an extensive study of algorithmic fairness. Numerous notions of fairness have been proposed, designed to capture different scenarios. These measures typically refer to a "protected group" in the data, defined using values of some sensitive attributes. Confirming whether a fairness definition holds for a given group is a simple task, but detecting groups that are treated unfairly by the algorithm may be computationally prohibitive as the number of possible groups is combinatorial. We present a method for detecting such groups efficiently for various fairness definitions. Our solution is implemented in a system called DENOUNCER, an interactive system that allows users to explore different fairness measures of a (trained) classifier for a given test data. We propose to demonstrate the usefulness of DENOUNCER using real-life data and illustrate the effectiveness of our method.

PVLDB Reference Format:
Jinyang Li, Yuval Moskovitch, and H. V. Jagadish. DENOUNCER: Detection of Unfairness in Classifiers. PVLDB, 14(12): 2719-2722, 2021.
doi:10.14778/3476311.3476328

1 INTRODUCTION

Automated data-driven decision-making tools are widely used in a vast range of application domains, from deciding who should get a loan [1], to automated hiring [2] and even in assessing the risk of paroling convicted criminals [5]. With the increasing use of data-driven tools, we also witness a large number of cases where these tools are biased or discriminate unfairly.

An article published recently in the New York Times [3] showcased this problem. The International Baccalaureate (IB) is a global standard of educational testing that allows U.S. high-school students to gain college credit. The final exams, which are a major factor in the students' scores, were canceled due to the COVID-19 pandemic. Instead, students were assigned grades based on a predictive model. As a result, high-achieving students from poor school districts were severely hurt, because the model placed great weight on school quality. For instance, students from low-income families were predicted to fail the Spanish exam, even when they were native Spanish speakers. Many of them had studied for IB hoping to save thousands of dollars on tuition by earning college credit with their scores.

Proceedings of the VLDB Endowment, Vol. 14, No. 12 ISSN 2150-8097.
doi:10.14778/3476311.3476328

#	Ethnicity	Zipcode	School ID	Spanish Exam	ML predication
1	White	103	9	Pass	Pass
2	**Hispanic**	**103**	**9**	**Pass**	**Fail**
3	White	103	10	Pass	Pass
4	**White**	**105**	**9**	**Fail**	**Pass**
5	White	103	9	Fail	Fail
6	Hispanic	105	9	Pass	Pass
7	White	105	10	Pass	Pass
8	**Hispanic**	**103**	**10**	**Pass**	**Fail**
9	Hispanic	103	10	Fail	Fail
10	White	105	9	Pass	Pass

Figure 1: Students' grades in Spanish exam. An Ethnicity of "White" is short hand for "Caucasian Non-Hispanic". The ML prediction is wrong for the highlighted rows.

The increasing impact of data-driven methods on society and their effect on human life, have given rise to increasing interest in the study of algorithmic fairness [9]. Various fairness measures have been proposed, where different measures are typically designed to capture case-appropriate properties: different definitions may be used in different use-cases. For instance, one natural fairness definition considers the accuracy among different groups. According to this definition, a classifier is fair if different groups in the data (i.e., Hispanic students from low-income families) have the same overall accuracy in prediction as other groups. Another plausible definition takes into account the false positive error rates. This definition is appropriate when minimizing the error of falsely classifying subject in the negative class as positive is desired (e.g., granting loans to people who would not be able to pay back). Deciding whether a classifier is fair depends on the notion of fairness. Different definitions can lead to different outcomes as we next demonstrate.

Example 1.1. Figure 1 shows a simplified dataset, inspired by the New York Times story about test score prediction. The dataset contains students' profile with information regarding their ethnicity, zipcode, and school. The "Spanish Exam" and "ML prediction" attributes depict the student's actual performance on the exam and the algorithm's prediction respectively. The overall accuracy of the model (based on the given data) is 0.7, however for Hispanic students the accuracy is only 0.5, significantly lower than any other group in the data (that may be defined by values of the different attributes). Another fairness measure, known as equal opportunity, focuses on the false negative error rate ($FNR = \frac{FN}{TP+FN}$). Intuitively, by this measure the classifier should give similar results for all students who pass the exam. In this case a high FNR indicates fairness issues. In this example Hispanic students have a high FNR (0.67) compared with the overall (0.29), but also students from zipcode 103 suffer from a relatively high FNR of 0.5.

Fairness definitions usually refer to a given "protected group" in the data, which is defined based on the values of some sensitive attributes (e.g., gender, race, age, or combinations thereof). When

developing a data-driven decision-making system, a data scientist may wish to ensure the algorithm is fair according to a given fairness measure. Given a protected group, confirming algorithmic fairness is a simple task. However the group definition may be unknown in advance. As a simple example, in the IB exams, one of the sources for bias in the results was the use of historical IB results of the schools. But using the school ID to define the protected group is not a natural or intuitive choice. When the group definition is unknown in advance, and can be defined by any value combination, searching for groups that are treated unfairly may be computationally prohibitive.

To this end, we propose an efficient method for detecting groups that are treated unfairly (according to a given definition) by a given classifier. Intuitively, we wish to avoid reporting "very specific" description of tuples that are mis-classified. Instead, we aim at discovering meaningful groups, namely, large enough groups, where the minimal group size is defined by the user. We have implemented our method in a system called DENOUNCER (for DEtectioN Of UNfairness in ClassifiERs). DENOUNCER allows the user to interactively explore different fairness measures of a trained classifier, with respect to a given test data. The system supports fairness definitions that are based on statistical properties of the predicted and actual outcome. It gets as input a classifier, test data, a fairness measure, and thresholds over the size of detected groups and the difference in their fairness measure compared to the overall value. DENOUNCER presents the user a concise description of each detected group along with their size and the selected fairness measure, and allows the user to explore individual tuple in the resulting groups. We will demonstrate the usefulness of DENOUNCER using three real-life datasets, allowing the audience to play the role of a data scientist.

Related work. The problem of detecting unfairness in machine learning models was studied in multiple lines of work. [8] proposes a statistical hypothesis test based on the theory of optimal transport to detect unfair classifiers. To intuitively show the bias in machine learning models, some visualization tools were proposed. FairSight [4] represents a workflow to support understanding, measuring, identifying and mitigating unfairness in decision making. FairVis [7] audits the fairness of ML models by finding similar discriminatory itemsets, and DiscriLens [10] facilities a better understanding and analysis of algorithmic discrimination while revealing the intertwining relationship among subgroups. These tools either focus on investigating only user-specific protected groups, or support individual fairness rather than group fairness.

2 TECHNICAL BACKGROUND

We (informally) introduce the main technical notions involved in the development of DENOUNCER using a running example. We assume the data is represented using a single relational database, and that the relation's attribute values are categorical. Where attribute values are drawn from a continuous domain, we render them categorical by bucketizing them into ranges: very commonly done in practice to present aggregate results. In fact, we may even group categorical attributes into fewer buckets where the number of individual categories is very large. We start by presenting our solution for the problem of detecting groups with low accuracy and then present a generalized solution for other fairness definitions.

2.1 Low accuracy detection

Given a database D with attributes $\mathcal{A} = \{A_1, \ldots, A_n\}$, we use $Dom(A_i)$ to denote the active domain of A_i (i.e., set of values appearing in the column A_i) for $i \in [1..n]$. A *pattern* p over D is a set of $\{A_{i_1} = a_1, \ldots, A_{i_k} = a_k\}$ where $\{A_{i_1}, \ldots, A_{i_k}\} \subseteq \mathcal{A}$ and $a_j \in Dom(A_{i_j})$ for each A_{i_j} in p. We say that a tuple $t \in D$ *satisfies* a pattern p and denote it by $t \vDash p$ if $t.A_{i_j} = a_j$ for each A_{i_j} appearing in p. The *size* $s_D(p)$ of a pattern p is then the number of tuples in D that satisfy p.

Example 2.1. Consider again the dataset given in Figure 1. p ={ Zipcode = 103, School ID = 9} is an example of a possible pattern. In the given data, the 1st, 2nd and 5th tuples satisfy p, thus $s_D(p) = 3$.

Given a classifier M and a labeled datast D, the *accuracy* of a pattern p, denoted as $acc_M(p)$, is the ratio of the number of tuples in D satisfying p that are correctly classified by M to $s_D(p)$.

Example 2.2. Consider again the dataset depicted in Figure 1 and the pattern p ={Zipcode = 103, School ID = 9} given in Example 2.1. As shown, tuples 1, 2 and 5 satisfy p, thus $s_D(p) = 3$. Since tuple 2 is misclassified, the accuracy of p is $\frac{2}{3}$.

Recall that our goal is to detect significant groups (i.e., large enough) that are treated unfairly by the algorithm (i.e., have low accuracy compared to the overall algorithm accuracy). In particular, we wish to provide the user with a concise description of these groups. Given a classifier M and an error threshold τ_a, we say that p is a *most general* pattern with accuracy below τ_a if $acc_M(p) \leqslant \tau_a$ and $\forall p' \subsetneq p$, $acc_M(p') \geqslant \tau_a$. Intuitively, such patterns are more significant in the data.

We are now ready to formally define our problem.

Problem 2.3 (Low Accuracy Patterns). *Given a database D, a trained classifier M with overall accuracy a, an accuracy delta threshold $\Delta\tau_a$, and a size threshold τ_s, find all most general patterns with size $\geqslant \tau_s$ and accuracy $\leqslant \tau_a$, where $\tau_a = a - \Delta\tau_a$.*

We next describe our algorithm for low accuracy patterns detection. The high level idea of the algorithm is as follows. Given a dataset D and a classifier M, we compute the set of misclassified tuples D^M_{mis} (tuples in D classified incorrectly by M). Note that given a pattern p, the accuracy of p can be computed as $acc_M(p) = 1 - \frac{s_{D^M_{mis}}(p)}{s_D(p)}$. We traverse the set of possible patterns (starting with the most general ones) and compute the accuracy for each pattern. To do so, we use the notion of *pattern graph* presented in [6]. Briefly, the nodes in the graph are the set of all possible patterns, and there is an edge between a pair of patterns p and p' if $p \subset p'$ and p' can be obtained from p by adding a single attribute value pair. In this case, we say that p (p') is a parent (child) of p' (p). As shown in [6], the pattern graph can be traversed in a top-down fashion, while generating each pattern at most once.

Algorithm. Given a labeled test dataset D, a classifier M and thresholds τ_s and $\Delta\tau_a$ over the size and accuracy of the pattern, we first use M and D to compute the set of misclassified tuples D^M_{mis} (tuples in D classified incorrectly by M) and $\tau_a = a - \Delta\tau_a$. We then traverse the pattern graph and compute the size of each pattern p in the given dataset D and in the misclassified dataset D^M_{mis}, and then use them to compute the pattern accuracy $acc_M(p)$.

Pruning. To optimize the search, we prune the search space as follows. First note that $s_D(p) \geqslant s_D(p')$ for every p and p' when p' is a descendent of p in the pattern graph. Thus, when reaching a pattern p with $s_D(p) < \tau_s$, the descendent of p can be pruned. Moreover, if $s_D(p) \geqslant \tau_s$ and $acc_M(p) \leqslant \tau_a$, its descendent can be pruned as well since we are looking for most general patterns. Finally, note that the accuracy of a pattern p is lower than τ_a if $acc_M(p) = 1 - \frac{s_{D^M_{mis}}(p)}{s_D(p)} \leqslant \tau_a$ i.e., $s_D(p)(1-\tau_a) \leqslant s_{D^M_{mis}}(p)$. Since we are interested only in patterns p with $s_D(p) \geqslant \tau_s$ we get $\tau_s \cdot (1-\tau_a) \leqslant s_{D^M_{mis}}(p)$. Thus, patterns with size less than $\tau_s \cdot (1-\tau_a)$ can be pruned as well.

Example 2.4. Consider again the dataset given in Figure 1 and the classifier M whose results are depicted in the "ML Prediction" column. Given the thresholds $\tau_s = 3$ and $\Delta\tau_a = 0.15$, the algorithm first computes τ_a and the set of misclassified tuples D^M_{mis}. In this case, as shown in Example 1.1 the overall accuracy is 0.7, thus $\tau_a = 0.55$ and $D^M_{mis} = \{2, 4, 8\}$. The most general patterns are p_1 ={Ethnicity=White}, p_2 ={Ethnicity=Hispanic}, p_3 ={Zipcode=103}, p_4 ={Zipcode=105}, p_5 ={School ID=9}, and p_6 ={School ID=10}. The algorithm first considers p_1. Since $s_{D^M_{mis}}(p_1) = 1 < (1-\tau_a) * \tau_s = 1.35$, this pattern and all of its descendants can be pruned. Next, the algorithm considers p_2. There are two Hispanic students with incorrect predicted grades, thus $s_{D^M_{mis}}(p_2) = 2 > 1.35$. Since the total number of Hispanic students in D is 4, the accuracy of p_2 is $acc_M(p_2) = 0.5 < \tau_a = 0.55$, so p_2 is added to the result set. p_3 is considered next. The accuracy of p_3 is $\frac{4}{6} = 0.67 > \tau_a$, thus its children p_7 = {Zipcode=103, School ID=9} and p_8 = {Zipcode=103, School ID=10} are generated. Similarly the children of p_4, p_5 and p_6 are generated (as none of them has low accuracy). In this example, the accuracy of all of these generated patterns is satisfactory, and the output of the algorithm is p_2 ={Ethnicity=Hispanic}. Note that the pattern {Ethnicity=Hispanic, zip=103} also has low accuracy but is not "most general", and thus not reported (and not even considered) by the algorithm.

2.2 Generalized solution

We now propose a generalization of the algorithm described in Section 2.1 to account for other definitions for algorithmic fairness [9]. In particular, we consider definitions that are based on the model's prediction and actual outcomes. These statistical measures of fairness rely on the possible cases of a classifier's outcome: True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN). We next provide a brief overview of such definitions[1] and refer the readers to [9] for more details.

Predictive parity The fraction of correct positive prediction $\frac{TP}{TP+FP}$ should be similar for all groups.

False positive error rate balance (predictive equality) The probability of a subject in the actual negative class to have a positive predictive value $FPR = \frac{FP}{FP+TN}$ is similar for all groups.

False negative error rate balance (equal opportunity) Similar to the above, but considers the probability of falsely classifying subject in the positive class as negative $FNR = \frac{FN}{TP+FN}$.

Equalized odds Combines the previous two definitions. All groups should have both similar false positive error rate balance $\frac{FP}{FP+TN}$ and false negative error rate balance $\frac{FN}{TP+FN}$.

Conditional use accuracy equality All groups should have similar probability of subjects to be accurately predicted as positive $\frac{TP}{TP+FP}$ and accurately predicted as negative $\frac{TN}{TN+FN}$.

Treatment equality This definition considers the ratio of error. A classifier satisfies it if all groups have similar ratio of false negatives and false positives.

Generalized algorithm. Our solution can be generalized to capture the aforementioned fairness definitions (and any definition that relies on the values TP, FP, FN and TN). The general algorithm is similar to the algorithm for detecting patterns with low accuracy with the following modifications. First, the algorithm computes four datasets (instead of D^M_{mis}): D^M_{TP}, D^M_{FP}, D^M_{TN} and D^M_{FN} that contains the tuples from D that are TP, FP, TN and FN respectively. The algorithm traverses the pattern graph in a top-down fashion. For each pattern it computes the size, and the fairness measure using D^M_{TP}, D^M_{FP}, D^M_{TN} and D^M_{FN}. We use τ_f to refer to the threshold of fairness value in the general case. Note that for some fairness measures (e.g., false positive/negative error rate balance), lower values are preferred. In this case the algorithm returns groups with fairness value higher than τ_f. Moreover, note that the pruning based on $s_{D^M_{mis}}$ presented in Section 2.1 is not applicable in general, however, other methods, such as pruning based on the nominator value, can be used.

Example 2.5. Consider again our running example. Assume we wish to use the false negative error rate balance as a fairness measure, and consider a case where the given thresholds are $\tau_s = 3$ and $\Delta\tau_f = 0.15$. D^M_{FN} consists of the tuples 2 and 8, and D^M_{TP} contains tuples 1, 3, 6, 7 and 10, thus, the false negative error rate balance of the model is $FNR = \frac{FN}{TP+FN} = \frac{2}{2+5} = 0.29$. Intuitively, individuals from groups with high FNR are more likely to get a false negative prediction, thus, in this case, we are interested in groups that have FNR $> \tau_f = 0.29 + \Delta\tau_f = 0.44$. Similarly to the detection of low accuracy patterns, the algorithm first considers the patterns $p_1, \ldots, p_6$ depicted in Example 2.4. $s_{D^M_{FN}}(p_1) = 0$, therefore the FNR of p_1 and its descendants is 0, and they can be pruned. The resulting groups in this example are p_2 = {Ethnicity=Hispanic} with FNR of 0.67 and p_3 ={Zipcode=103} with FNR of 0.50.

We note that the algorithm in [6] alone is not sufficient for our needs, and some additional steps are required, such as extracting subsets of the dataset (e.g., D^M_{mis}) and traversing them (in addition to the given datasets) in parallel in order to compute the fairness measures. This difference provides us with new pruning opportunities that were not applicable in [6].

3 SYSTEM OVERVIEW

DENOUNCER is implemented in Python 3 and runs on macOS Big Sur. The user interface is implemented in Python 3 with PySimpleGUI package. The general architecture of DENOUNCER is shown in Figure 2. We next briefly explain the components of the system.

The inputs to the system are: a labeled test dataset D, a trained classifier M , a fairness measure f, and the two thresholds: τ_s, over the output groups' sizes, and $\Delta\tau_f$, over the delta from their

[1]The definitions given in [9] consider two groups (protected/unprotected groups defined using a sensitive attribute). We generalize the definitions to fit our problem of detecting unfairly treated groups.

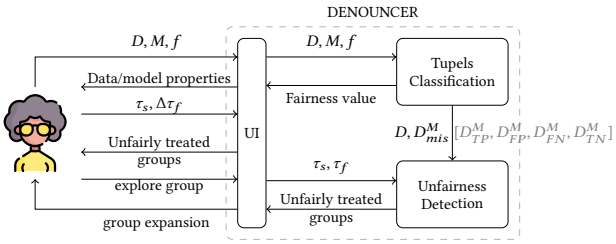

Figure 2: System architecture

overall fairness value. The user can either select a dataset from previously loaded datasets or provide a '.csv' file. Upon selection, DENOUNCER previews the user part of dataset's content and its size. Similarly, users can load a classifier (provided as a Python pickled file) or select one from a predefined set. The user can then select a fairness measure from a drop-down menu. DENOUNCER also allows the user to restrict the set of attributes the algorithm should consider in the computation (i.e., select a subset of potential sensitive attributes).

Once the dataset, the classifier and the fairness measure are set, DENOUNCER starts with the tuple classification phase. In this step, the system uses M and D to generate D^M_{mis} (in the case of overall accuracy) or D^M_{TP}, D^M_{FP}, D^M_{TN} and D^M_{FN} (if other fairness measure was selected), and computes the overall fairness value for the given fairness measure f. The overall fairness value is presented to the user, and can be used to determine $\Delta\tau_f$. Given $\Delta\tau_f$, DENOUNCER uses it to compute τ_f which is used in the group detection phase.

The output of the tuple classification step, along with the input data D and the thresholds τ_s and τ_f, are used as input for the unfairness detection phase, where the pattern traversal part of the algorithm is implemented. Finally, the unfairly treated groups along with their sizes and fairness values are presented to the user. DENOUNCER allows further exploration of the resulting groups. By clicking on an output group, the user can view the tuples in the group with their true and predicted values.

4 DEMONSTRATION SCENARIO

We will demonstrate the usefulness of DENOUNCER in detecting groups that suffer from algorithmic unfairness with respect to a given classifier, fairness definition and test data. For the demonstration we will use three real-life datasets.

- The COMPAS dataset[2] that was collected and published by ProPublica as part of their investigation into racial bias in criminal risk assessment software. It contains demographics, recidivism scores produced by the COMPAS software, and criminal offense information for 6,889 individuals.
- The Default of Credit Card Clients Dataset[3], which contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.
- The Adult dataset[4] with information about individual's annual income based on the census data. It was collected and used to explore the possibility in predicting income level (over/under $50K$ a year) based on the individual's personal information. The dataset contains information such as age, gender, race, workclass, education level, marital status and occupation.

The audience will be asked to play the role of a data scientist, examining the benefits of DENOUNCER in the analysis of a classifier. We will let participants interactively explore the fairness measures of different classifiers with respect to each of the datasets, and the resulting unfairly treated groups.

In the first part of the demonstration we will ask the audience to select a dataset and a classifier. For the demonstration we will use 3 pre-defined classifiers: decision tree, random forest and Ada boost, each one trained over each of the aforementioned datasets. We will start with the overall accuracy fairness definition. Upon selection of the dataset and classifier we will ask the user to set the size and fairness thresholds (compared to the overall accuracy) based on the data size and the model's accuracy. We will let the audience explore the tuples in the resulting groups, and then ask them to vary the thresholds and observe the effect on the results.

Finally, we will consider the effect of different fairness definitions on the results. Participants will be asked to select a fairness measure among the measures depicted in Section 2.2. We will illustrate the diverse outcomes in each scenario. For example, we will compare the resulting groups when using false positive error rate balance and false negative error rate. It has famously been shown in the context of the COMPAS dataset that multiple fairness objectives cannot be satisfied simultaneously. Participants will be able to see this for themselves, and develop an intuition for the issues, as they tune the knobs in DENOUNCER.

ACKNOWLEDGMENTS

This work was supported by NSF grants 1741022 and 1934565.

REFERENCES

[1] 2011. The Algorithm That Beats Your Bank Manager. https://www.forbes.com/sites/parmyolson/2011/03/15/the-algorithm-that-beats-your-bank-manager/?sh=4fbafadc1ae9.
[2] 2015. Can an Algorithm Hire Better Than a Human? https://www.nytimes.com/2015/06/26/upshot/can-an-algorithm-hire-better-than-a-human.html.
[3] 2020. When Algorithms Give Real Students Imaginary Grades. https://www.nytimes.com/2020/09/08/opinion/international-baccalaureate-algorithm-grades.html.
[4] Yongsu Ahn and Yu-Ru Lin. 2019. Fairsight: Visual analytics for fairness in decision making. *IEEE transactions on visualization and computer graphics* 26, 1 (2019).
[5] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. 2016. Machine bias. *ProPublica, May* 23, 2016 (2016).
[6] Abolfazl Asudeh, Zhongjun Jin, and H. V. Jagadish. 2019. Assessing and Remedying Coverage for a Given Dataset. In *ICDE*.
[7] Ángel Alexander Cabrera, Will Epperson, Fred Hohman, Minsuk Kahng, Jamie Morgenstern, and Duen Horng Chau. 2019. Fairvis: Visual analytics for discovering intersectional bias in machine learning. In *VAST*.
[8] Bahar Taskesen, Jose Blanchet, Daniel Kuhn, and Viet Anh Nguyen. 2020. A Statistical Test for Probabilistic Fairness. *arXiv preprint arXiv:2012.04800* (2020).
[9] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *FairWare*.
[10] Qianwen Wang, Zhenhua Xu, Zhutian Chen, Yong Wang, Shixia Liu, and Huamin Qu. 2020. Visual analysis of discrimination in machine learning. *IEEE Transactions on Visualization and Computer Graphics* (2020).

[2] https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis
[3] https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients
[4] https://archive.ics.uci.edu/ml/datasets/adult