# A short account of FAIR-DB: a system to discover Data Bias

(Discussion Paper)

Fabio Azzalini[1,2], Chiara Criscuolo[1] and Letizia Tanca[1]

[1]Politecnico di Milano - Dipartimento di Elettronica, Informazione e Bioingegneria
[2]Human Technopole - Center for Analysis, Decisions and Society

**Abstract**

Computers and algorithms are increasingly pervading our daily lives, therefore to trust these systems we have to make sure that the data they use are fair and without bias. As a result, Fairness has become a relevant topic of discussion within the field of Data Science, and technologies that accurately discover discrimination and bias present in datasets are of paramount importance.

In this work we present FAIR-DB (FunctionAl dependencIes to discoveR Data Bias), a novel framework to detect biases and discover discrimination in datasets. By exploiting various kinds of functional dependencies, our tool can identify those attributes in a database that encompass discrimination (e.g. gender, ethnicity or religion) and the ones that instead satisfy various fairness criteria.

We compared our framework with two state-of-the-art systems for detecting unfairness in datasets, obtaining overall similar results on a real-world dataset; specifically, the comparison highlighted that FAIR-DB not only provides very precise information about the groups treated unequally, but also that, in comparison with other existing tools, may obtain more insights regarding the bias present in datasets

**Keywords**

Fairness, Data Bias, Functional Dependencies

## 1. Introduction

In the recent years fairness has become an important topic of interest in the Data Science community. Indeed, computers and algorithms have made our lives efficient and easier, but among the prices we risk to pay is the possible presence of discrimination and unfairness in the decisions we make with their support [1]. Data Science technologies are based on data, and for them to be reliable we have to make sure that the data we feed them are fair and without bias: *data can be considered of good quality only if they conform to high ethical standard* [1]: to avoid (possibly unintentional) unethical behaviors and their consequences, data cleaning tools should also include *tools to discover bias in data* [2].

In this paper we present *FAIR-DB* [3], a framework that, by discovering and analyzing particular types of functional dependencies, can find unfair behaviors in a dataset and guide its correction. A *Functional Dependency* ($FD : X \rightarrow Y$) is a class of database integrity constraints that hold

*SEBD 2021: The 29th Italian Symposium on Advanced Database Systems, September 5-9, 2021, Pizzo Calabro (VV), Italy*
fabio.azzalini@polimi.it (F. Azzalini); chiara.criscuolo@polimi.it (C. Criscuolo); letizia.tanca@polimi.it (L. Tanca)
0000-0003-0631-2120 (F. Azzalini); 0000-0003-2607-3171 (L. Tanca)


[1]https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

between two sets $X$ and $Y$ of attributes in a relation of a database. It specifies that the values of the attributes of $X$ uniquely (or *functionally*) determine the values of the attributes of $Y$ [4]. In a FD, X is called antecedent or left-hand-side (LHS) while Y is called consequent, or right-hand-side (RHS). In Conditional Functional Dependencies (or CFDs) [5], conditions are used to specify the subset of tuples on which a dependency holds: a CFD is a pair $\left(X \rightarrow Y, t_p\right)$, where $X \rightarrow Y$ is a standard functional dependency and $t_p$ is a *pattern tuple* over the attributes in $X$ and $Y$; for each $A$ in $X \cup Y$, $t_p[A]$ is a constant *'a'* in *dom(A)*, or an unnamed variable *'_'*.
In this paper we consider Approximate Conditional Functional Dependencies (ACFDs), i.e., uncertain CFDs that hold only on a subset of the tuples, to detect biases and discover discrimination in the datasets subject to analysis, by recognizing cases where the value of a certain attribute (e.g. gender, ethnicity or religion) *frequently determines* the value of another one (such as range of the proposed salary or social state).

## 2. State of the Art

Most of the research done in the area of Data Science Ethics is carried out by the Machine Learning community; three possible approaches can be adopted when trying to enforce fairness in a data analysis application: *(i) preprocessing techniques*, i.e. procedures that, before the application of a prediction algorithm, make sure that the learning data are fair; *(ii) inprocessing techniques*, i.e. procedures that ensure that, during the learning phase, the algorithm does not pick up the bias present in the data, and *(iii) postprocessing techniques*, i.e. procedures that correct the algorithm's decisions with the scope of making them fair.
One of the first preprocessing techniques was presented by Pedreschi et al. [6]: exploiting the concept of association rules and custom metrics, the system can identify potentially discriminatory itemsets.
A project that provides the implementation of many interesting techniques is *AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias* [7], this work presents a new open-source framework whose aim is to reach algorithmic fairness. The system tries to mitigate algorithmic bias by exploiting techniques that use statistical measures to compute fairness.
In the Machine Learning context, the majority of works that try to enforce fairness are related to a prediction task, and more specifically to classification algorithms in decision-making tools [8], [9], [7]. The main difference between these approaches and our framework, that solves unfairness adopting a *preprocessing technique*, is that our system does not need a classifier to work, because it is based on finding conditions (in form of approximate constraints) that are already present in the data, even though possibly with some level of approximation. Furthermore, building a classifier to solve the fairness problem requires the policy to be application-oriented, greatly limiting the applicability of these systems to scenarios where other tasks are needed.
A very interesting work on fairness in Machine Learning that employs a *preprocessing technique* is *Nutritional Labels for Data and Models* by Stoyanovich and Howe [10]. The authors developed an interpretability and transparency tool based on the concept of *Nutritional Labels*, drawing an analogy to the food industry, where simple and standardized labels convey information about the ingredients and the production processes. Nutritional labels are derived semi-automatically

as part of the complex process that gave rise to the data or model they describe. The final system is called *Ranking Facts*, and automatically derives nutritional labels for ranking. *Ranking Facts* is a collection of visual widgets, in particular, the *Fairness* widget quantifies whether the ranked output exhibit statistical parity (a particular definition of group fairness) with respect to one or more protected attributes.

## 3. Methodology

Before presenting the methodology we first introduce some fundamental notions that will accompany us along our discussion. Given a dataset $D$, the *support* of a CFD $\left(X \rightarrow Y, t_p\right)$ is defined as the proportion of tuples $t$ in the dataset $D$ which contain $t_p$, that is:

$$Support(X \rightarrow Y, t_p) = \frac{|t \in D; t_p \subseteq t|}{|D|}$$

The *confidence* is an indication of how often the CFD has been found to be true. Let $t_p = (x \cup y)$ where $x$ is a tuple over $X$ and $y$ is a tuple over $Y$. The confidence value of a CFD $\left(X \rightarrow Y, t_p\right)$ is the proportion of the tuples $t$ containing $x$ which also contain $y$:

$$Confidence(X \rightarrow Y, t_p) = \frac{|t \in D; t_p \subseteq t|}{|t \in D; x \subseteq t|}$$

*FAIR-DB* is composed by the following main steps: *Data Preparation and Exploration*, *ACFD Discovery and Filtering*, *ACFDs Selection*, *ACFDs Ranking* and *ACFDs User Selection and Scoring*. In the next subsections we present each phase in detail, with the help of the following example:

**Example 1. Running Example** *For our experiments, we have used the U.S. Census Adult Dataset*[2] *[11], containing information about many social factors of US adults, like 'Income', 'Age', 'Workclass', 'Education', 'Education-Num' (i.e. the number of years already attended at school), 'Marital-Status', 'Race', 'Sex', (work) 'Hours-Per-Week', 'Native-Country', and some more.*

### 3.1. Data Preparation and Exploration

In this phase we import the data, perform (if needed) data integration and apply the typical data preprocessing steps (solve missing values, apply discretization etc.) needed to clean and prepare the data. As a last step of this first phase, *Data Visualization* can be of great help in analyzing the characteristics of the data; in fact, from the plots, a user can understand whether groups are present in the dataset and more specifically, if there is a majority class for a certain attribute, and if present can identify minorities.
This preliminary phase gives a general idea of the dataset, and during this step we can guesstimate the protected columns and identify, if present, the target variable of our study. Regarding the running example, we selected 'Sex', 'Race' and 'Native-Country' as protected attributes and 'Income' as target variable.

[2]https://archive.ics.uci.edu/ml/datasets/Adult

| |
|---|
| (Education-Degree='Middle-school') → Income='≤ 50K' |
| (Age-Range='15-30') → Income='≤ 50K' |
| (Education-Degree, Income='≤ 50K') → Native-Country |
| (Native-Country='NC-Hispanic') → Income='≤ 50K' |
| (Income='≤ 50K') → Native-Country='NC-US' |

**Table 1**
CFD Discovery output

| ACFD | Support | Difference |
|---|---|---|
| (Native-Country = 'NC-Hispanic') → (Income = '≤50K') | 0.0439 | 0.1570 |
| (Sex = 'Female') → (Income = '≤50K') | 0.2874 | 0.1352 |
| (Race = 'Black') → (Income = '≤50K') | 0.0813 | 0.1190 |

**Table 2**
A few of the selected ACFDs using the *Difference metric*

## 3.2. ACFD Discovery and Filtering

In this phase we extract the ACFDs from the dataset, using the algorithm of [12]. The algorithm expects as input the following three parameters: *the (minimum) support threshold*, the *(minimum) confidence threshold,* and the size of the largest antecedent *maxSize*.
Given an instance $D$ of a schema $R$, support threshold $\delta$, confidence threshold $\epsilon$, and maximum antecedent size $\alpha$, the approximate CFDs discovery problem is to find all ACFDs $\phi$: $\left(X \rightarrow Y, t_p\right)$ over $R$ with: support$(\phi, D) \geq \delta$, confidence$(\phi, D) \geq \epsilon$ and $|X| \leq \alpha$. The ACFDs obtained are in this form: $(lhsAttr1 = v1, ..., lhsAttrN = vN) \rightarrow (rhsAttr = v)$.
The algorithm returns all the dependencies that satisfy the constraints.

**Example 2. CFD Discovery** *The algorithm, applied to the dataset resulting from the previous phase, finds 118 ACFDs. Table 1 reports a few of them.*
Now we filter the dependencies, discarding the ones that do not satisfy two constraints:

- all the attributes must be assigned a value (the ACFDs that contains only constants in both the LHS and RHS are called *Constant ACFDs*[13]);
- at least one protected attribute and the target variable have to be present inside the dependency, so that the ACFDs might show bias.

**Example 3. ACFDs Filtering** *From Table 1 we discarded the third dependency for not being a Constant ACFD and the first three dependencies for not containing any protected attribute. After this phase we are left with 84 of the original 118 dependencies.*

## 3.3. ACFD Selection

This phase is responsible for finding the dependencies that *actually reveal* unfairness in the dataset. To do so, we have devised an unfairness measure, called *Difference*, that indicates how much a dependency is *'unethical'*. The higher the difference, the more the ACFD reveals

an unfair behavior. In order to assess the unfair behavior of a dependency, we also take into consideration its support, that indicates the *pervasiveness* of the ACFD; unethical dependencies with high support will impact many tuples, and thus will be more important.
We define the *Difference* of a dependency $\phi$ as the difference between the confidence of $\phi$ and the confidence of the dependency computed without the protected attributes of the LHS of the ACFD. Let $\phi : (X \rightarrow Y, t_p)$ be an ACFD, and $Z = (X - \{ProtectedAttributes\})$, that is, the LHS of the dependency without its *protected attributes*, and let $z$ be the restriction of $t$ to $Z$. Then:

$$\text{Difference}(\phi) = \text{Confidence}(\phi) - \text{NoProtectedAttributeConfidence}(\phi)$$

where

$$\text{NoProtectedAttributeConfidence}(\phi) = \frac{|t \in D; t_p \subseteq t|}{|t \in D; z \subseteq t|}$$

That is:

$$\text{Difference}(\phi) = \frac{|t \in D; t_p \subseteq t|}{|t \in D; x \subseteq t|} - \frac{|t \in D; t_p \subseteq t|}{|t \in D; z \subseteq t|}$$

The *Difference* metric gives us an idea of how much the values of the protected attributes influence the value of $Y$.

**Example 4. Difference score of a dependency** *Analyzing the following dependency:* $\phi_1 : (Sex =` Female', Workclass =` Private') \rightarrow (Income =`\leq 50K')$ *we can compute the* Difference *as:* $Diff(\phi_1) : Conf(\phi_1) - Conf(\phi_1')$*, where* $\phi_1' : (Workclass =` Private') \rightarrow (Income =`\leq 50K')$*. Three different behaviors can emerge:*

- *If the Difference is close to zero, fairness is respected since it means that females are treated equally to all the elements of the population that have the same characteristics (without specifying the protected attribute).*
- *If the Difference is positive it means that the women that work privately and gain less than 50K dollars/year are overall treated worse than the generality of people that work privately and gain less than 50K dollars/year.*
- *If the Difference is negative the opposite situation is detected.*

Finally, we choose the ACFDs whose *Difference* is above the *minThreshold*, which means that there is a significant inequality between the group involved in the ACFD and the general behaviour of the population.

**Example 5. Selected ACFDs** *Table 2 reports three of the seventeen dependencies that satisfied the selection criteria along with their relevant metrics. From the example, "Hispanic", "Female" and "Black" groups suffer from discrimination with respect to the rest of the population, in fact, people that belong to one or more of these groups have an income that is below the 50000 dollars/year because of their nationality, sex or race.*

### 3.4. ACFD Ranking

In a real-world dataset, the number of ACFDs selected in the previous step could be very large, therefore for the user to look at all these dependencies would be a very demanding task to complete. Thus, it is necessary to order the dependencies according to some criterion, enabling the user to analyze the most important and interesting ones first, speeding up the process and reducing the cost. In our framework the user can order the dependencies according to one of the following criteria:

- *Support-based*: the support indicates the proportion of tuples impacted by the dependency; the higher the support, the more tuples are involved by the ACFD. Ordering dependencies by support highlights the *pervasiveness* of the dependency.
- *Difference-based*: this criterion highlights the dependencies where the values of the protected attributes influence most the value of their RHS, therefore this ordering privileges the *unethical aspect* of the dependencies.
- *Mean-based*: this method tries to combine both aspects of a dependency: the unethical perspective and its pervasiveness. Sorting the ACFDs using this criterion results in positioning first the dependencies that have the best trade-off between difference and support.

### 3.5. ACFD User Selection and Scoring

In this last phase the user selects from the ranked list $N$ dependencies that are interesting for the research needs. Using only the $N$ selected ACFDs, the framework computes two scores that summarize the properties of the entire dataset:

- *Cumulative Support*: is the percentage of tuples in the dataset involved by the selected ACFDs. The more this value is close to 1, the more tuples are impacted by unfair dependencies.
- *Difference Mean*: is the mean of all the *'Difference'* scores of the selected ACFDs. It indicates how much the dataset is unethical according to the dependencies selected. The greater the value, the higher the bias in the dataset.

**Example 6. ACFDs User Selection and Scoring** *The user chooses N = 15 ACFDs that are interesting according to her needs among the dependencies obtained after the ranking step. The total number of tuples involved by the ACFDs is 13296 while the total number of tuples in the dataset is 30169; this results in a Cumulative Support of 0.44. The Difference Mean is 0.16. These two scores indicate that a considerable number of tuples, 44%, show a behavior that is very different, on average 16%, from the fair one. Finally, a deeper analysis of the dependencies confirm that the dataset is unfair with respect to all the protected attributes; the groups more discriminated are: 'Female', 'Black', 'NC-Hispanic' and 'Amer-Indian-Eskimo'. Table 3 reports a few interesting ACFDs.*

| |
|---|
| (Sex = 'Female') → Income = '≤ 50K' |
| (Race = 'Black') → Income = '≤ 50K' |
| (Race = 'Amer-Indian-Eskimo') → Income = '≤ 50K' |
| (Native-Country = 'NC-Hispanic') → Income = '≤ 50K' |

**Table 3**
A few user-selected dependencies from the U.S. Census Adult Income Dataset

# 4. Experimental Results

We now present the results obtained by *FAIR-DB* on a real-world dataset, and a comparison between our framework and the competitors presented in the Introduction chapter. The dataset we considered is the *U.S. Census Adult Income Dataset*, already briefly presented in Example 1. The version we considered contains 32561 tuples and 13 attributes, 5 of which are numerical and 8 are categorical.

## 4.1. *FAIR-DB* Results

We recall from Example 6 the results of *FAIR-DB* on the U.S. Census Adult Income Dataset, which scores a *Cumulative Support* of 0.44 and a *Difference Mean* of 0.16, indicating that many tuples show an unfair behavior. Specifically, the dataset highlights bias towards all the protected attributes: 'Sex', 'Race' and 'Native-Country'; a deeper analysis confirms that the most discriminated groups are: 'Female', 'Black', 'NC-Hispanic' and 'Amer-Indian-Eskimo'.

## 4.2. Comparison with competitors

The results obtained by *Ranking Facts*[10] and with *AI Fairness 360*[7] are in complete accordance with ours. *Ranking Facts* finds in the U.S. Census Adult Income Dataset unfair behaviors across all the three protected attributes with discrimination against: 'Female', 'Black', 'NC-Hispanic' and 'Amer-Indian-Eskimo'.
*AI Fairness 360* analyzed fairness through statistical measures computed for two protected attributes: 'Race' and 'Sex' checking fairness only for one binary attribute at the time. For the most of statistical measures, the 'Race' attribute has a privileged group composed by 'White' people and a discriminated one composed by 'Not White' people, the 'Sex' attribute has a privileged group composed by 'Male' and a discriminated one composed by 'Female'.
The two competitors check fairness property only for one binary attribute at the time, while, since the ACFD technique can involve more than one attribute at a time, our tool can report information about subgroups fairness, actually detecting unfair behaviors at finer level of granularity; as a results *Ranking Facts* and *AI Fairness 360* do not contain information about the existing bias in minorities. This is very important, because the discrimination might be limited to some specific scenario (e.g. not all the women but only the black women working in the private sector), and this information is very useful to guide the phase of DB repair.
Finally, we present a theoretical comparison between *FAIR-DB* and the work by Pedreschi et al. [6]. Their system extracts association rules that signal potentially discriminatory itemsets. The process exploits a specific type of association rules that constrains the target class to be only on the RHS, as a result, the potentially discriminatory itemset can be only on the LHS; in

our approach the ACFDs found do not have this constraint. Finally, [6] does not involve user interaction, therefore the user cannot discard the rules that are not interesting for the specific investigation. Moreover, FAIR-DB provides as a final step a set of summarizing metrics that describes the overall degree of unfairness of the dataset.

## 5. Conclusion and Future Works

We presented *FAIR-DB*, a novel framework, that, through the extraction of a particular type of Functional Dependencies, can discover bias and discrimination present in a datasets.
Future works will include:, *(i)* the addition to the system of a dependency repair phase, that, starting from the selected ACFDs will correct the dataset removing all the unfair behaviors from it *(ii)* the study of dependencies with high confidence and low support to highlight interesting, not necessarily frequent, behaviors, *(iii)* the development of a graphical user interface to facilitate the interaction of the user with the system, *(iv)* the study of other classes of functional dependencies[5], *(v)* a deeper comparison with *Ranking Facts* and other similar methods.

## References


[1] D. Firmani, L. Tanca, R. Torlone, Ethical dimensions for data quality, Journal of Data and Information Quality (JDIQ) (2019).

[2] J. Stoyanovich, B. Howe, H. Jagadish, Responsible data management, Proceedings of the VLDB Endowment (2020).

[3] F. Azzalini, C. Criscuolo, L. Tanca, FAIR-DB: Functional dependencies to discover data bias, Workshop Proceedings of the EDBT/ICDT 2021 (2021).

[4] S. Abiteboul, R. Hull, V. Vianu, Foundations of Databases, Addison-Wesley, 1995. URL: http://webdam.inria.fr/Alice/.

[5] L. Caruccio, V. Deufemia, G. Polese, Relaxed functional dependencies—a survey of approaches, IEEE Transactions on Knowledge and Data Engineering (2015).

[6] S. Ruggieri, D. Pedreschi, F. Turini, Data mining for discrimination discovery, ACM Transactions on Knowledge Discovery from Data (TKDD) 4 (2010) 1–40.

[7] R. K. Bellamy, K. Dey, M. Hind, S. C. Hoffman, S. Houde, K. Kannan, P. Lohia, J. Martino, S. Mehta, A. Mojsilović, et al., AI Fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias, in: IBM Journal of Research and Development, IBM, 2019.

[8] J. A. Adebayo, et al., FairML: ToolBox for diagnosing bias in predictive modeling, Ph.D. thesis, Massachusetts Institute of Technology, 2016.

[9] F. Tramer, V. Atlidakis, R. Geambasu, D. Hsu, J.-P. Hubaux, M. Humbert, A. Juels, H. Lin, Fairtest: Discovering unwarranted associations in data-driven applications, 2017 IEEE European Symposium on Security and Privacy (2017).

[10] J. Stoyanovich, B. Howe, Nutritional labels for data and models., IEEE Data Eng. Bull. (2019).

[11] D. Dua, C. Graff, UCI machine learning repository, 2017. URL: http://archive.ics.uci.edu/ml.

[12] J. Rammelaere, F. Geerts, Revisiting conditional functional dependency discovery: Splitting the "C" from the "FD", Springer, 2018.

[13] W. Fan, F. Geerts, J. Li, M. Xiong, Discovering conditional functional dependencies, IEEE Transactions on Knowledge and Data Engineering (2010).