# Fairness-Aware Instrumentation of Preprocessing Pipelines for Machine Learning*

Ke Yang, Biao Huang
New York University
ky630,bh1918@nyu.edu

Julia Stoyanovich
New York University
stoyanovich@nyu.edu

Sebastian Schelter
University of Amsterdam
s.schelter@uva.nl

## ABSTRACT

Surfacing and mitigating bias in ML pipelines is a complex topic, with a dire need to provide system-level support to data scientists. Humans should be empowered to debug these pipelines, in order to control for bias and to improve data quality and representativeness.

We propose *fair-DAGs*, an open-source library that extracts directed acyclic graph (DAG) representations of the data flow in preprocessing pipelines for ML. The library subsequently instruments the pipelines with tracing and visualization code to capture changes in data distributions and identify distortions with respect to protected group membership as the data travels through the pipeline. We illustrate the utility of *fair-DAGs* with experiments on publicly available ML pipelines.

**ACM Reference Format:**
Ke Yang, Biao Huang, Julia Stoyanovich, Sebastian Schelter. 2020. Fairness-Aware Instrumentation of Preprocessing Pipelines for Machine Learning. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3398730.3399194

## 1 INTRODUCTION

Software systems that learn from data to automate decisions are being deployed in increasing numbers in public and private sector application scenarios. In recent years, various issues with respect to fairness, accountability and transparency in such systems have been raised [2, 9]. In contrast to textbook machine learning (ML) scenarios—building a classifier over a clean and static numeric input dataset—real-world end-to-end ML applications are often very complex [10], especially in terms of the preprocessing operations applied to the training data [14]. These pipelines often *integrate data from multiple input sources*, subsequently *apply data cleaning operations* such as missing value imputation and filtering, and *encode the raw data as features* using various techniques. This paper focuses on surfacing bias in such complex pipelines.

Data preprocessing operations can introduce bias in subtle ways: (*i*) Filters and joins can reduce the number of records in the data and thereby distort the proportion of protected (minority or historically disadvantaged) groups *even if* they do not directly use the protected attribute as a join key or target predicate. (*ii*) Simple methods for missing value imputation may also also distort protected group proportions. For example, consider an application that gives users a binary choice of gender and also allows to leave gender unspecified. Then, the gender column will have missing values, which a data scientist may attempt to fill in using *mode imputation*, thus setting all unspecified gender values to either male or female, depending on the most frequent demographic in the data. (*iii*) More generally, multi-class classification for missing value imputation typically only uses the most frequent classes as target variables [3], leading to a distortion for small protected groups, because membership in these groups will never be imputed. (*iv*) Another source of bias could be the usage of pretrained word embeddings. For example, a pipeline may replace a textual `name` attribute with the corresponding vector from a word embedding that is missing for rare, non-western names. If we then filter out records for which no embedding was found, this filter may disproportionately remove members of protected groups.

According to a classification of bias in computer systems [7], points (*i*)- (*iv*) illustrate *technical bias* that arises due to the design of data cleaning methods. Further, these systems are subject to *preexisting bias* "that exists independently, and usually prior to the creation of the system" [7]. An example is that survey data collected from ethnic minorities may contain more missing values and it may be noisier than data collected from the majority ethnic group [8]. To make matters even more challenging, we are often interested in quantifying and mitigating bias with respect to multiple protected groups, such as gender *and* ethnicity, either independently

*This work was supported in part by NSF Grants No. 1926250 and 1934464, and by the Moore-Sloan Data Science Environment at New York University.


*HILDA'20, June 14–19, 2020, Portland, OR, USA*

ACM ISBN 978-1-4503-8022-5/20/06…$15.00
https://doi.org/10.1145/3398730.3399194

or in combination. The latter is important because discriminatory effects may be amplified for individuals who belong to multiple disadvantaged groups — a phenomenon known as intersectionality in the social sciences [5].

To identify the introduction of such bias, we propose *fair-DAGs*, an open-source library that empowers data scientists to inspect and debug ML preprocessing pipelines, in order to control for bias and to improve data quality and representativeness. We make the following contributions.

- We propose the *fair-DAGs* library, which integrates into existing ML applications and extracts a directed acyclic graph (DAG) representation of the preprocessing operations and the flow of data.
- We discuss how to instrument the pipeline with tracing code (based on the DAG representation) to capture data distributions and identify distortions with respect to protected groups as the data flows through the pipeline.
- We illustrate the utility of *fair-DAGs* with experiments on publicly available preprocessing pipelines.

The *fair-DAGs* library is available at https://github.com/DataResponsibly/fairDAGs, and the Web UI can be accessed at https://dataresponsibly.github.io/tools.

## 2 APPROACH

Our approach is based on the idea of leveraging a logical representation of the operators and the data flow in a preprocessing pipeline for ML. Such a representation allows us to automatically reason about programs (similar to query plans for relational databases), and enable *automated tracing* by instrumenting the code to record changes in the proportion of records from protected groups. We instrument operators such as joins, filters and missing value imputers that may change these proportions.

**Pipeline representation**. We focus on ML pipelines that apply common declarative abstractions [13] such as pandas data slicing operations or scikit-learn's `ColumnTransformer`[1] to define feature transformations. These pipelines have a natural representation as directed acyclic graphs (DAGs) [13]. Our library extracts a DAG, in which vertices represent operations such as filters and joins, and edges denote the flow of data in the pipeline. In our case, the data sources hold relational data, while the data flowing through the DAG either comprises of collections of relational tuples or tensors. We support relational operators including join, selection, and projection (consuming and producing relational data), standard feature encoders like one-hot-encoders (consuming relational data and outputting matrices), and ML preprocessing operations like standardization (consuming and producing matrices).

[1]https://scikit-learn.org/stable/modules/generated/sklearn.compose.ColumnTransformer.html

**Implementation**. Our *fair-DAGs* library takes a Python script that implements an ML pipeline as input, along with user-defined annotations that denote the target variable of the ML model, and the categorical attributes that label members of legally protected groups. In this work, we target linear ML pipelines without control flow that leverage pandas and scikit-learn for data preprocessing. As observed in [11], such linear code accounts for more than 87% of all cells in data science notebooks, and our chosen libraries occur in 30% to 50% of notebooks. We derive the DAG representation by inspecting the Abstract Syntax Trees (AST) of the Python code of the preprocessing pipeline, where we identify common operations from pandas such as data loading via `pd.read_csv`, relational joins via `pd.join`, or scikit-learn's `Pipeline` data structure. We apply code instrumentation for tracing the data at runtime via the `inspect` module[2], by profiling the input and output data of selected operators.

**Example**. Figure 1 illustrates our approach. Its left side shows Python code for an ML pipeline that prepares data for a credit assessment model. It reads historical financial performance data of the applicants and their demographics, imputes missing values, performs feature encoding and transformation, and finally trains a logistic regression model to estimate credit worthiness. The right side of Figure 1 shows our extracted DAG representation, which allows us to identify operators that potentially modify the proportions of protected groups in the data and thus may introduce bias (circled in purple). The *fair-DAGs* library subsequently instruments the pipeline with tracing code, based on our extracted representation. In the example in Figure 1, we trace the categorical attributes `gender` and `age` (the latter stands for "age group"), present the corresponding data distributions, and show the proportion of positively classified instances for each value of these attributes. When a data scientist executes the pipeline, the system stores the intermediate data generated after each operation and records the changes in the target feature for all sensitive attribute groups. This information is then presented to the data scientist, empowering her to understand the impact of individual operations on the performance for different demographic groups, and react to problems.

## 3 EVALUATION

In our experimental evaluation of *fair-DAGs* we pursue three goals: (1) to ascertain the feasibility of extracting DAG-based representations in ML pipelines created by practitioners; (2) to explore how filtering and data cleaning might lead to distortions with respect to protected group membership; and (3) to compare performance of pipelines that process the same datasets and have the same goal, but use different preprocessing operations, for intersectional groups.

[2]https://docs.python.org/3/library/inspect.html

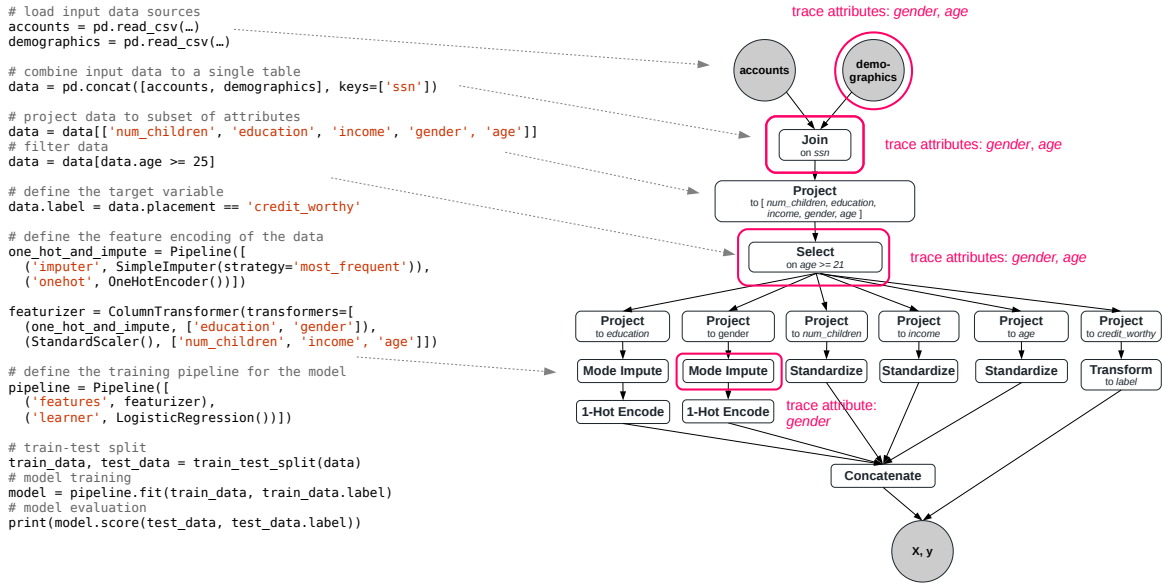


**Figure 1: Instrumentation of a pandas/scikit-learn based ML pipeline via a DAG representation of the data flow.**

**Experimental setup**. We conduct experiments on two benchmark datasets that are frequently used in the algorithmic fairness literature. *Adult income*[3] contains information about 33,000 individuals from the 1994 U.S. census, with sensitive attributes `gender` and `race`. The corresponding task is to predict whether the annual income of an individual exceeds $50,000. Gender bias has been reported for this task, with males having a higher probability of the positive outcome (income greater than $50,000). Pipeline $P1$ was used to preprocess *Adult income* in [6, 12]:

```
1 raw_data = pd.read_csv(f_path, na_values='?')
2 data = raw_data.dropna()
3 labels = binarize(data['income-per-year'], ['>50K'])
4 feature_transformation = ColumnTransformer([
5   ('cat', OneHotEncoder(), ['education', 'workclass']),
6   ('num', StandardScaler(), ['age', 'hours-per-week'])])
7 income_pipeline = Pipeline([
8   ('features', feature_transformation),
9   ('classifier', DecisionTreeClassifier())])
```

Note that $P1$ removes records with missing values in line 2. Pipeline $P2$ denotes a modified pipeline on *Adult income*, which imputes missing values rather than removing them:

```
1 categorical_feature_transformation = Pipeline([
2   ('impute', SimpleImputer(strategy='most_frequent')),
3   ('encode', OneHotEncoder(handle_unknown='ignore'))])
```

The *COMPAS* dataset contains information about 6,889 criminal defendants in Broward County, FL, along with predictions of their recidivism risk, as produced by a commercial tool called COMPAS. The sensitive attributes include `gender` and `race`. The task is to predict whether a defendant is likely re-offend. Racial discrimination has been reported by ProPublica [1], quantified as the disparity in false-positive rates (FPR, higher for Blacks) and false-negative rates (FNR, higher for Whites). Pipeline $P3$ on *COMPAS* filters data according to the rules outlined in [1]:

```
1  df = pd.read_csv(f_path, na_values='N/A')
2  data = df[(df.days_b_screening_arrest <= 30)
3    & (df.days_b_screening_arrest >= -30) & (df.is_recid != -1)
4    & (df.c_charge_degree != 'O') & (df.score_text != 'N/A')]
5  data = data.replace('Medium', 'Low')
6  labels = LabelEncoder().fit_transform(data['score_text'])
7  impute1_and_onehot = Pipeline([
8    ('imputer1', SimpleImputer(strategy='most_frequent')),
9    ('onehot', OneHotEncoder())])
10 impute2_and_bin = Pipeline([
11   ('imputer2', SimpleImputer(strategy='mean')),
12   ('disc', KBinsDiscretizer(n_bins=4, enc='ordinal'))])
13 featurizer = ColumnTransformer([
14   ('impute1_and_onehot', impute1_and_onehot, ['is_recid']),
15   ('impute2_and_bin', impute2_and_bin, ['age'])])
16 pipeline = Pipeline([
17   ('features', featurizer),
18   ('classifier', LogisticRegression())])
```

**Extracted representations**. Figures 2, 3, and 4 show the DAGs of $P1$, $P2$ and $P3$ generated by *fair-DAGs*. For an operation in a pipeline, a node is extracted and circled with a color-coded boundary that corresponds to the color in the code listing. Note that scikit-learn operations can be nested, such as lines 4-5 in 3 and lines 5-8 in 3, and we only color the corresponding attributes on which the operations focus.

**Results**. We execute the pipelines and leverage *fair-DAGs* to automatically compute statistics about the distortion of the data with respect to protected groups. *fair-DAGs* produced detailed reports. We now discuss the main findings available to data scientists from these reports.

[3] http://archive.ics.uci.edu/ml/datasets/Adult

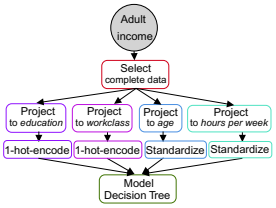


**Figure 2: DAG of $P1$ on *Adult income***

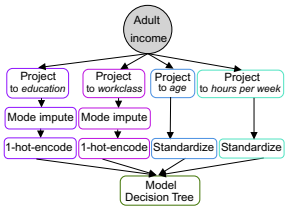


**Figure 3: DAG of $P2$ on *Adult income***

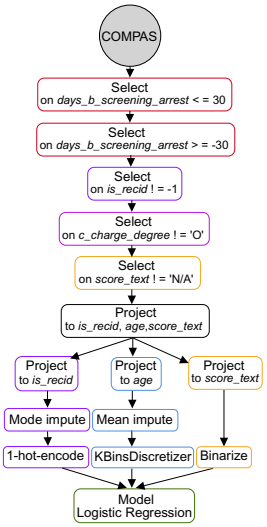


**Figure 4: DAG of $P3$ on *COMPAS***

*Impact of data cleaning.* We evaluate how the data cleaning operation in $P1$ — removing all tuples with missing values, shown by the red node in Figure 2 and the red line in the code listing, affect the distortions in sensitive attributes. This cleaning operation removes 1,647 tuples, or 7.2% of the dataset. 234 of these correspond to individuals who have an annual income of $> 50K\$$ (i.e., about 4.3% of such individuals are removed). We investigate the impact of data cleaning on the sensitive attributes `gender` and `race`. We observe that this cleaning operation removes two times more females (6%) than males (3.9%) among the individuals with a positive label (annual income $> 50K\$$) in the training data. *fair-DAGs* also reports group specific classification metrics and allows us to determine that the under-representation of females is accompanied by a higher FPR in the ML model learned from this training set. We additionally find that non-White females are removed in higher proportion than White females (e.g., 17.1 % of low income females with *Asian-Pac-Islander* as `race` vs. only 8.9% of low income White females), pointing to an intersectional issue. Our library helps with identifying such issues by visualising histograms of distortions for each operation and histograms of performance measures (e.g. FPR and selection rate) for the ML model in a pipeline.

*Impact of filter operations.* Another preprocessing operation that may introduce bias is filtering according to user-determined rules. We now evaluate how the filtering operations in pipeline $P3$ on *COMPAS* affect the distortions in sensitive attributes. We investigate the changes of predicted risk score caused by filtering on the attribute `days before screening arrest`, represented by the the red node in Figure 4 and by red attributes in the code listing for $P3$. *fair-DAGs* calculates the selection rate of being labeled high risk (i.e., the probability to be predicted likely to recidivate by the ML model from $P3$ on the test set). *fair-DAGs* allows us to verify that the outlined filtering operation removes data relatively evenly among gender groups, and among intersectional groups on gender and race, even though males (18.5%) are filtered somewhat more frequently than females (16.7%) among defendants who have high risk scores.

**Summary**. We demonstrate the *fair-DAGs* can extract DAG representations and instrument typical preprocessing pipelines used in fairness research. Our library helps identify problematic distortions in sensitive attributes and in classification outcomes introduced by preprocessing operations, such as the increased under-representation of females (and of Black females) in *Adult income*, accompanied by a higher FPR.

**Future Work**. In future work, we aim to extend *fair-DAGs* to pipelines with control flow, support a larger set of preprocessing frameworks (e.g., Tensorflow Transform[4]), reduce the performance overhead of tracing, and integrate techniques for automated model validation on data slices [4].

## REFERENCES

[1] Julia Angwin et al. 2016. Machine bias: There's software used across the country to predict future criminals. *And it's biased against blacks. ProPublica* 23 (2016).
[2] Solon Barocas and Andrew D Selbst. 2016. Big data's disparate impact. *Calif. L. Rev.* 104 (2016), 671.
[3] Felix Biessmann et al. 2018. Deep Learning for Missing Value Imputation in Tables with Non-Numerical Data. *CIKM* (2018).
[4] Yeounoh Chung et al. 2018. Slice finder: Automated data slicing for model interpretability. *SysML Conference* (2018).
[5] Kimberle Crenshaw. 1989. Demarginalizing the intersection of race and sex: A black feminist critique of antidiscrimination doctrine, feminist theory and antiracist politics. *u. Chi. Legal f.* (1989), 139.
[6] Sorelle A. Friedler et al. 2019. A Comparative Study of Fairness-enhancing Interventions in Machine Learning. *FAT** (2019).
[7] Batya Friedman and Helen Nissenbaum. 1996. Bias in computer systems. *TOIS* 14, 3 (1996), 330–347.
[8] Joost Kappelhof. 2017. Survey research and the quality of survey data among ethnic minorities. *Total survey error in practice* (2017), 235–252.
[9] Cathy O'Neil. 2017. *Weapons of math destruction: How big data increases inequality and threatens democracy.* Broadway Books.
[10] Neoklis Polyzotis et al. 2017. Data management challenges in production machine learning. *SIGMOD* (2017), 1723–1726.
[11] Fotis Psallidas et al. 2019. Data Science through the looking glass and what we found there. *CoRR* abs/1912.09536 (2019).
[12] Babak Salimi et al. 2019. Interventional Fairness: Causal Database Repair for Algorithmic Fairness. *SIGMOD* (2019).
[13] Sebastian Schelter et al. 2017. Automatically tracking metadata and provenance of machine learning experiments. *Machine Learning Systems Workshop at NeurIPS* (2017).
[14] Sebastian Schelter et al. 2020. FairPrep: Promoting Data to a First-Class Citizen in Studies on Fairness-Enhancing Interventions. In *EDBT*.

[4]https://www.tensorflow.org/tfx/transform/