Fairness in Data Wrangling

Lacramioara Mazilu, Norman W. Paton, Nikolaos Konstantinou, Alvaro A.A. Fernandes
School of Computer Science, University of Manchester, Manchester, United Kingdom
lara.mazilu, norman.paton, nikolaos.konstantinou, alvaro.a.fernandes @manchester.ac.uk

***Abstract*—At the core of many data analysis processes lies the challenge of properly gathering and transforming data. This problem is known as *data wrangling*, and it can become even more challenging if the data sources that need to be transformed are heterogeneous and autonomous, i.e., have different origins, and if the output is meant to be used as a training dataset, thus, making it paramount for the dataset to be *fair*. Given the rise in usage of artificial intelligence (AI) systems for a variety of domains, it is necessary to take into account fairness issues while building these systems. In this paper, we aim to bridge the gap between gathering the data and making the datasets fair by proposing a method for performing data wrangling while considering fairness. To this end, our method comprises a data wrangling pipeline whose behaviour can be adjusted through a set of parameters. Based on the fairness metrics run on the output datasets, the system plans a set of data wrangling interventions with the aim of lowering the bias in the output dataset. The system uses Tabu Search to explore the space of candidate interventions. In this paper we consider two potential sources of dataset bias: those arising from unequal representation of sensitive groups and those arising from hidden biases through proxies for sensitive attributes. The approach is evaluated empirically.**

***Index Terms*—data wrangling, fairness, bias, sample size disparity, proxy attribute, training dataset**

I. Introduction

Machine learning supports decision making. However, machine learning is often crucially dependent on the available training data, and it is important that the training data does not manifest biases that may lead to inappropriate learned models. Bias may lead to the inconsistent treatment of subgroups within the population, for example in relation to race or gender. Reducing bias, and thus increasing fairness, has become a topic for widespread investigation, with proposals that intervene before, during and after the learning process [9].

Although these interventions to reduce bias close to and within the learning process are important, it is clear that data scientists are spending a significant fraction of their time on data wrangling [10], the process of selecting, combining and cleaning the available data sets, and it seems likely that decisions made during data wrangling could lead to bias. However, existing work on bias reduction has not investigated the potential impact of data wrangling, or how to reduce the likely bias associated with a wrangling process.

In this paper, we investigate how a data wrangling process can be revised to: *(i)* identify features during wrangling that may give rise to bias; and *(ii)* automatically modify the wrangling process with a view to reducing the bias in the result of the wrangling process.

More specifically, we consider two dataset features that may be associated with bias: *sample size disparity* and *proxy attributes*. Sample size disparity refers to inconsistent levels of representation of different groups in data sets, and proxy attributes are those that implicitly represent a sensitive group (e.g., *neighborhood* could be a proxy attribute for *race*). Given a sensitive attribute, such as gender or race, it is straightforward to detect *a posteriori* the presence of sample size disparity and proxy attributes. However, this suggests the possibility that if data wrangling can be carried out differently, potential sources of bias could be significantly reduced, thereby reducing the onus on the data analyst to detect and remove them.

In this paper, we build on the VADA approach [18] to automate data wrangling processes such that given a description of a target structure, the system searches for ways to populate the target from a set of sources. Given this capability, when biases are detected in the generated wrangling plan, we identify ways of intervening in the wrangling process that may help to reduce the bias. The contributions of this paper are:

1) A characterization of problems that may arise during data wrangling and lead to bias in downstream analyses.
2) A proposal for how to detect such problems and respond to them in a system that automates the generation of data wrangling pipelines.
3) A search algorithm that explores alternative ways of wrangling the data with a view to producing fair results from data wrangling.
4) An evaluation of (3) for a benchmark data set that has been used to study fairness interventions.

The remainder of this paper is structured as follows. Section II situates our work in relation to other work on obtaining fair outcomes before, during or after training machine learning models. Section III describes the data wrangling pipeline used. Section IV gives an overview on the proposed approach, i.e., how we revise the generation of a data wrangling process with a view to obtaining fairer datasets. Section V provides the details of the proposed approach in terms of assessing fairness, and identification of suitable collections of interventions in the data wrangling process. In Section VI, the system is evaluated with respect to fairness in the output data on a set of scenarios obtained using a real-world biased dataset. Section VII concludes.

II. Related Work

This section reviews related results that seek to increase fairness in machine learning. Proposals have been made to intervene before, during and after the learning process, so here we discuss how biases are detected and responded to in each case. As our work considers interventions before learning, we particularly focus on pre-processing data for analysis. There are several ways to tackle the problem of removing the bias from a model development cycle, thus, given the extensive on-going work on the topic, the list of techniques we mention below is an incomplete list of all available algorithms.

Pre-processing. Techniques that pre-process the data aim to transform the available dataset so that the bias is removed [6]. The most straightforward technique is through suppression, where the sensitive attribute and all related to it are removed from the dataset [13]. Another method proposes relabelling the tuples in the dataset so that the bias is removed which usually comes with the challenge of selecting the tuples to be relabelled [12], [13]. For instance, in [12], a Naive Bayesian classifier is used for selecting the tuples to be relabelled. In [22] a technique is proposed that finds a way to hide the information of the sensitive attribute by encoding the data. Another method that proposes alterations on the attributes with the effect of removing disparate effect is [8].

There are also techniques that do not modify the initial data. One of them is reweighing, which adds extra-information to the dataset by assigning weights to the tuples [3], [13]. However, not all models can make use of the additional information, thus, other proposals have focused on keeping the data as is through resampling [21]. Techniques include adding synthetic data, oversampling [4], [17], and removing tuples, undersampling [5].

In-processing. In-processing techniques represent alterations on the learning algorithms in order to address the problem of having biased outcomes during the model training step [6]. In [15], [22] a regularization term is proposed that is aware of potential discrimination. In [23] a method is proposed that aims to train a classifier to maximize the prediction accuracy and, at the same time, minimize the ability of an adversary to determine the sensitive attribute values based on predictions.

Post-processing. Post-processing techniques are applied after the model is trained, without intervening on the training phase. In [11] a method is proposed as a post-learning step that changes the outcome labels such that the bias is minimized w.r.t. equalized odds [11]. The technique proposed in [14] changes the output labels by giving positive labels to unfavoured groups and negative to favoured.

III. Technical Background

This paper builds on an existing data wrangling system [18] that heretofore has wrangled data without considering the fairness of the results. This section provides the details required to understand the interventions described in Section V.

Figure 1(a) shows the data wrangling pipeline together with its expected input, i.e., a target schema, a set of sources, and a set of parameters. The following are the steps in the data wrangling process that we will refer to further later.

Schema matching. Schema matching generates source-to-target relationships that express semantic correspondences between the source attributes and the target attributes. In this paper, we consider matches of the following form:

$m_{ST} : S.a \rightarrow T.a'$, where S is an input source relation, $a \in schema(S)$ is an attribute in S; T is a target relation and $a' \in schema(T)$ is an attribute in T. The defined match expresses that the values from source attribute $S.a$ can be transferred in the target attribute $T.a'$ as they are semantically equivalent.

Data profiling. Before mapping generation, profiling infers candidate keys among the source attributes and inclusion dependencies between pairs of source attributes. The profile data is defined as follows:

candidate keys – a column (or a combination thereof) that has unique values in the relation in which it occurs;

(partial) inclusion dependencies – given two projections R and S with identical arity over relations R' and S', resp., we define the inclusion dependency $R \subseteq_{\theta_{R,S}} S$, where $\theta_{R,S}$ represents the overlap of values between attributes R and S, i.e., the ratio of distinct values from R included in the values of S.

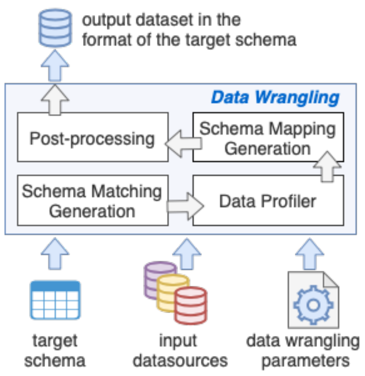
Mapping generation. The mapping generation component [20] creates mappings over the pool of input sources by performing a search through the space of possible mappings using a dynamic programming approach where the inference of relationships between the sources is done using profile data. The candidate keys and the (partial) inclusion dependencies are used to infer (approximate) foreign keys between sources coming from different origins, i.e., sources that do not have explicit join paths declared.

Post-processing. The post-processing component ensures that the output dataset does not contain subsumed tuples, i.e., redundant data. We consider a tuple to be subsumed if it satisfies the conditions of *subsumed tuples* as defined in [2]: *A tuple $t_1 \in T$, where T is a relation, subsumes another tuple $t_2 \in T$ if: **(i)** t_1 and t_2 have the same schema, **(ii)** t_2 contains more null values than t_1, and **(iii)** t_2 coincides in all non-null attribute values with t_1.*

IV. Approach Overview

The problem of generating unbiased datasets addressed in this paper can be defined as: *Given a collection of source tables S, a target schema T with an indicated protected attribute P, and (at least) one fairness metric, generate a dataset in the format of the target schema T such that the data therein is obtained from the input data sources S and the computed bias is minimised w.r.t. the chosen fairness metric.*

We address the above defined problem through an approach that uses a data wrangling pipeline (described in Section III - Figure 1(a)) in an exploration technique that relies on the *tabu search* paradigm. Figure 1(b) shows a high-level overview on the system that integrates the wrangling pipeline together with the search steps. In Figure 1(b), the first step is to run the data wrangling process without considering any fairness metric. This is considered the initialization phase. Then, the system checks if the stopping criteria is met. We describe our used stopping criterion in Section V-D. If the system decides that the search can continue, then the dataset is assessed based on the fairness metrics. We describe how we compute the bias values in Section V-A. If the dataset is considered to be biased, then a set of intervention plans is created in order to steer


output dataset in the format of the target schema
Data Wrangling
Post-processing
Schema Mapping Generation
Schema Matching Generation
Data Profiler
target schema
input datasources
data wrangling parameters
(a) Data wrangling pipeline

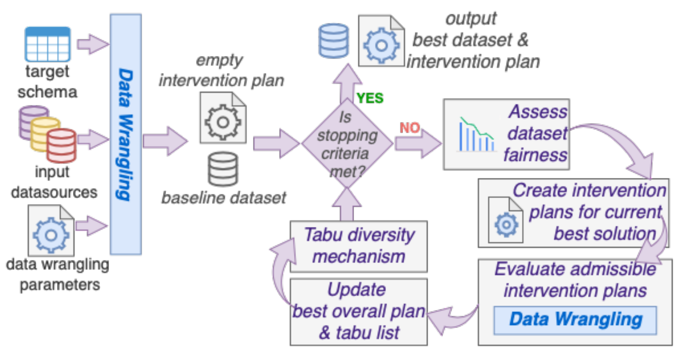

target schema
input datasources
data wrangling parameters
Data Wrangling
empty intervention plan
baseline dataset
output best dataset & intervention plan
YES
Is stopping criteria met?
NO
Assess dataset fairness
Create intervention plans for current best solution
Evaluate admissible intervention plans
Data Wrangling
Update best overall plan & tabu list
Tabu diversity mechanism
(b) Finding wrangling programs w.r.t. fairness

Fig. 1. Data wrangling search system considering fairness

the wrangling process. We describe the possible interventions, i.e., the components of an intervention plan in Section V-B. The wrangling process is steered through the set of input parameters that are specified based on the intervention plan. After the plans are created, the system evaluates non-tabu plans (Section V-D), i.e., runs the data wrangling process with their corresponding parameters. After all plans are evaluated, using an objective function (Section V-C), the system updates the *best overall plan* (if possible) and chooses the *best local plan* to explore further. Also, all explored intervention plans from this iteration are added to the tabu list so that the system does not explore them again in subsequent runs. Before it recycles, the system checks through the diversity mechanism if it is stuck in a local optimum. If it is, then it jumps to another search area. We describe our used diversity mechanism in Section V-D.

V. Approach

In this section we describe in more detail the architecture for finding *fair* datasets in a data wrangling scenario.

A. Assessing Fairness

Although the focus of our work is to aid building fair training datasets, the two metrics that we have chosen to focus on can be computed without the existence of a class label (i.e., as required for building classification models). Below we outline the two fairness metrics that we consider here.

Sample size disparity. Sample size disparity (*underrepresentation* in [1]) occurs when the proportion of tuples in the training data that do (and do not) belong to one of the sensitive groups leads to loss of accuracy in outcomes (e.g., an over-frequent decision to hire males because of fewer female tuples in the training set used to build the hire-or-not classifier). We compute how *fair* a dataset T is w.r.t *sample size disparity*. Let r_u (Eq. 1) be the ratio of values of the unfavoured group (u) in the target table T with the sensitive attribute S.

$$r_u = \frac{|\{t \in T, t.S = u\}|}{|T|} \tag{1}$$

We consider the dataset T to be *fair* if $r_u \in [r_d - e, r_d + e]$, where r_d is the desired ratio of tuples containing the value of the unfavoured group (u) on the sensitive attribute (S), and e is the acceptable error. The bias b_{SSD} (Eq. 2) is the difference between the desired ratio (r_d) and the obtained ratio (r_u):

$$b_{SSD} = abs(r_d - r_u) \tag{2}$$

Example 1: Assume a scenario where a model is trained to predict hiring likelihood of a person, with the sensitive binary attribute *gender*, i.e., with two groups, *male* and *female*. If the *male* group appears in significantly more tuples than the *female* group, then a model trained on such a dataset can specialize better at predicting outcomes for *male* entries than for *female* entries. In this case, in order to achieve high accuracy for both groups, an increase in the proportion of *female* entries may be needed to avoid bias in the model.

Proxy attributes. The presence of proxy attributes leads to hidden bias when a value in the sensitive attribute can be determined from the value of another attribute, which thereby acts as a proxy for the sensitive one [16].

In this paper, we determine a value dependency by considering approximate functional dependencies [19] between the sensitive attribute and other dataset attributes. We compute how *fair* a dataset T is w.r.t hidden bias through *proxy attributes* of the sensitive attribute S as follows.

Consider an approximate functional dependency *afd* of the form $P \rightarrow S$ where S is the sensitive attribute, and P is an attribute in the same table as S, $P \neq S$. The approximate functional dependency coefficient $c_{P\rightarrow S}$ denotes the ratio of tuples that satisfy the *afd*. Correspondingly, $1 - c_{P\rightarrow S}$ denotes the ratio of tuples that violate it.

Given a set of approximate functional dependency coefficients $C = \{c_{P\rightarrow S} | P, S \in schema(T), \}$, we compute the bias for *proxy attributes* as defined in Eq.3.

$$b_{PA} = \frac{\sum_i (c_i^2)}{\sum_i (c_i)} \tag{3}$$

where $c_i \in C$. b_{PA} is the weighted average of the coefficients of detected approximate functional dependencies. The weights for the coefficients are the coefficients themselves because, this way, the algorithm can assign them proportional penalties. We consider a dataset T as being biased if $b_{PA} > t$, where t is a given threshold.

Example 2: Consider the same scenario in Example 1. Assume that before training, the *gender* attribute is deleted from the dataset. Further assume that an attribute *marital status* exists whose domain is {*Husband, Wife, Divorced, Single*}. Thus, for the most part, when a *Wife* value appears, the *gender* attribute has value *female*. Likewise for the (*Husband*, *male*) pair. In such a case, gender comes very close to being inferrable from the *marital-status* attribute, even if *gender* itself is removed from the dataset.

B. Available Interventions

As explained in Section III, a data wrangling process uses matches and profiling data to perform informed decisions when deciding on how to transform data in the sources into the format stipulated by the user for the target. Note that the wrangling pipeline takes as input a set of parameters. If these parameters are changed, then the outcome of the wrangling process will differ in ways that ultimately stem from the new information. In order to adjust the inputs to the wrangling process we consider interventions on two types of parameter: *matches* and *inclusion dependencies*. Each of these interventions can have different effects on the generated wrangling process, which we outline below.

Matches. A match informs the system about the data that can be transferred from a source attribute to the target. In our data wrangling pipeline, the matching component finds all source attributes that can contribute to a target attribute. As an intervention, we can consider the exclusion of a match from the pipeline. The removal of a match $m_{ST} : S.a \rightarrow T.a'$ can have one of the following effects:

1) The matching component in the wrangling pipeline may find another source attribute $S.a''$, in the same source S, such that $m_{ST} : S.a'' \rightarrow T.a'$. In other words, $S.a''$ is detected as being another way that source S can contribute to the target T.
2) The matching component in the wrangling pipeline may find another source attribute $P.a$, in another source P, $P \neq S$, such that $m_{PT} : P.a \rightarrow T.a'$. In other words, $P.a$ is an alternative to $S.a$, therefore it can contribute to $T.a'$ with information from P replacing source S.
3) The matching component may not find an alternative source attribute, which potentially means populating attribute $T.a'$ with *null* values.

The system decides to exclude a match if the (biased) output dataset contains a sensitive or a proxy attribute that was populated using that match.

Inclusion dependencies. An inclusion dependency informs the system about overlapping values across source attributes. This information is used to infer inter-source join opportunities. In our data wrangling pipeline, the mapping generation component finds all combinations for merging the sources based on the profiling data, including inclusion dependencies. The removal of an inclusion dependency $I_{P,S} = P \subseteq_{\theta_{P,S}} S$, where $\theta_{P,S}$ is the degree of overlap, can have one of the effects:

1) The mapping generation component may find another join opportunity between the same two sources, S and P, so that they are still joined albeit on a different pair of attributes.
2) The mapping generation component may not find another way to join the two sources directly, but they may still be joined through a third source that shares a join attribute with both sources.
3) The mapping generation component may not find another way to join the two sources, which therefore results in their data no longer being aligned.

The system decides to exclude an inclusion dependency if the (biased) output dataset was built based on a merge that resulted from the information brought by the inclusion dependency, e.g., a join resulted from the overlap information.

Intervention plan. The intervention plan is a set of matches and/or inclusion dependencies that are excluded from the wrangling pipeline, i.e., not used to create the target data.

Example 3: Continuing Example 2, for a target $T(ms, edu, gender)$, consider the source $S(id, ms, name)$ with a match $m_1 : S.ms \rightarrow T.ms$ on *marital status* (ms) to the target T. S was joined with another source $P(id, edu, gender)$ on attributes id as they had an inclusion dependency $I_{P,S} = P.id \subseteq_\theta S.id$. The output dataset resulted as biased: $T.ms$ is a proxy attribute for $T.gender$ (see Example 2). Given that $T.ms$ was populated by $S.ms$, the system can try to remove the (proxy) bias through one of the following plans: ***(i)*** $plan_1 = \{m_1\}$, ***(ii)*** $plan_2 = \{I_{P,S}\}$, or ***(iii)*** $plan_3 = \{m_1, I_{P,S}\}$.

C. Objective Function

In order to compare different intervention plans, we use an objective function. Given that an intervention that aims to minimize the bias can result in a less complete dataset, we use an objective function that takes into consideration the trade off between bias and data completeness. For this, we use a weighted sum where weights add up to 1, $\sum w_i = 1.0$. The *cost* of an intervention plan is described by the objective function in Eq. 4. The overall aim is to minimize the *cost*, thus, less costly plans are preferred to more costly ones.

$$cost = \sum_{i=0}^{\#strategies} w_i * b_i + w_Q * r_{nulls} \tag{4}$$

where *(i)* b_i (see Eq.1, 2, and 3) abstracts over the computed bias for each of the *active* fairness metrics (sample size disparity and/or proxy attributes), *(ii)* r_{nulls} is the ratio of missing values in the evaluated dataset, and *(iii)* w_i and w_Q are the weights of the bias(es) and of the completeness.

D. Searching the Space of Potential Wrangling Programs

Search algorithm. We use Tabu Search to find an effective solution in the search space by iteratively moving from one potential solution to a better solution, until the stopping criteria is satisfied. Search algorithms can become stuck in local optima or in areas where the scores plateau. Tabu Search explores the neighborhood of a candidate solution and chooses the best candidate to explore next. In order to avoid becoming stuck in a local optimum, Tabu Search is able to jump out of a local optimum by including a diversity property that allows the search to move to another part of the search space such that it has a chance of discovering better candidate solutions. Tabu Search precludes the exploration of already-visited solutions using a *tabu* list containing points in the search space that must not be explored again, i.e., that were *tabooed*.

Tabu Search for our problem. We explore the search space of possible intervention plans that could be applied to the wrangling process in order to produce a less biased output dataset. In order to apply Tabu Search to our problem, we need to map concepts in our problem domain to search concepts, viz., neighbourhood, tabu list, stopping criterion, and approach to diversification. We now describe how we have done that.

Neighbourhood generation. When considering a candidate intervention plan, the data wrangling pipeline is run with the interventions stipulated in the plan. Based on the output of the data wrangling process, if the resulting dataset is still biased w.r.t. the fairness strategy, a new set of interventions is then produced. The neighbourhood for the currently explored plan is built by considering the resulting new interventions. A neighbour is a new intervention plan comprising all interventions in the currently explored plan together with one intervention from the resulting new interventions. However, among the new plans (i.e., neighbours), some may be *tabooed* and therefore will be discarded rather than explored.

Tabu list. Tabu Search proposes an efficiency strategy based on the use of a tabu list, which contains points in the search space that have been explored. The purpose of a tabu list is to prevent the exploration of intervention plans that have already been evaluated (more on plan evaluation in *Finding potential wrangling programs*).

Diversity mechanism. Tabu Search uses a diversity mechanism to jump out from an area that could have reached a local optimum. For our problem, a diversity mechanism is run if the *overall best plan* has not changed after a certain number of explored neighbourhoods. This number is a user-defined threshold and is given as input to Algorithm 1 (see below).

Jumping out is implemented by randomly choosing a subset of intervention plans where all plans are grouped by the number of interventions they contain. Then, from the randomly-selected subset of plans, the algorithm chooses the one with the lowest cost (computed by the objective function). The algorithm checks if the chosen plan has already been explored before. If it has not, it proceeds to explore it. If it has, it chooses the next best plan in terms of cost. If all plans have already been explored, then the algorithm tries to jump to another area of the search space by randomly selecting another subset of plans and repeats the same procedure until a plan to be explored is found, or a maximum number of attempts is reached, in which case it desists.

Stopping criterion. Given that the exploration space can potentially become too large to explore in its entirety, the search algorithm needs a stopping criterion. The algorithm stops the search if the upper limit for the number of evaluated plans has been reached. The upper limit is a user-defined number that is given as input in Algorithm 1.

Algorithm 1 Initialization

```
1: function INIT(datasets[], t, maxEvPlans, maxUnchangedBestPlan)
2:     $tabuList \leftarrow []$ //global variable
3:     $evaluatedPlans = 0$ //global variable
4:     $uBP \leftarrow maxUnchangedBestPlan$ //global variable
5:     $bestPlan \leftarrow \perp$ //global variable
6:     $noPlanDS, I, b, c \leftarrow$ EVALPLAN$(datasets, t, bestPlan)$
7:     $bestCost \leftarrow$ OBJFUNCTION$(b, c)$ //global variable
8:     EXPLORENEIGHBOURHOOD$(datasets, t, plan, I)$
9:     return $bestPlan$
```

Algorithm 2 Evaluate Plan

```
1: function EVALPLAN(datasets[], t, plan)
2:     $outputDS, I \leftarrow$ DATAWRANGLING$(datasets, t, plan)$
3:     $bias \leftarrow$ COMPUTEBIAS$(outputDS)$
4:     $cost \leftarrow$ COMPUTECOST$(outputDS)$
5:     $tabuList \leftarrow tabuList \cup \{plan\}$
6:     $evaluatedPlans \leftarrow evaluatedPlans + 1$
7:     return $outputDS, I, bias, cost$
```

Finding potential wrangling programs. The system starts searching for suitable wrangling programs that could reduce the bias. Below we outline the main system components. We call a wrangling program a wrangling pipeline that runs under a set of parameters. In our case, the set of parameters is given by the *intervention plan* (as defined in Section V-B).

INIT. Before starting the search, the algorithm needs a base case from which to start. This is done in the initialization method INIT shown in Algorithm 1. INIT takes as input a set of datasets, a target schema t, a number of maximum evaluated plans, $maxEvPlans$, and a number representing the maximum number of explored neighbourhoods without changing/updating the *best overall plan*. On line 5, the initialization *best overall plan* is a plan with no interventions. The search is started by evaluating the empty plan (line 6). Here, the EVAL method returns four variables: the dataset resulting from the wrangling ($noPlanDS$), the set of interventions that could be taken into consideration for next plans (I), the bias (b) and the cost for dataset quality (c). This initialization step represents the step where the system correlated the source datasets and did not aim to minimize the bias in the output dataset, thus, becoming the *baseline* case. In line 8, the search starts by exploring the neighbourhood of the baseline case.

EVALPLAN. The evaluation of a plan means running the wrangling process for it and then evaluating the resulting output. EVALPLAN is outlined in Algorithm 2 that takes as input the set of input data sources, the target and the intervention plan that needs to be evaluated. On line 2, through the DATAWRANGLING method, the system performs the wrangling for the sources and the target, where the wrangling parameters are updated according to the plan, i.e., the parameters contain, for exclusion, matches and/or inclusion dependencies according to the input plan. DATAWRANGLING outputs the resulting dataset

Algorithm 3 Neighbourhood Exploration

```
1: function EXPLORENEIGHBOURHOOD(datasets[], t, exlPlan, I)
2:     if CHECKSTOPCRITERIA() then
3:         return
4:     $bestLocalPlan \leftarrow \perp$
5:     $bestLocalCost \leftarrow inf$
6:     $nextI \leftarrow []$
7:     $neighbours \leftarrow$ GENNEIGHBOURS($exlPlan, I$)
8:     for each $neighbour$ in $neighbours$ do
9:         if CHECKSTOPCRITERIA() then
10:            break
11:        if $neighbour$ not in $tabuList$ then
12:            $nDS, nI, nB, nC \leftarrow$ EVALPLAN($datasets, t, neighbour$)
13:            $nCost \leftarrow$ OBJFUNCTION($nB, nC$)
14:            if $nCost < bestLocalCost$ then
15:                $bestLocalPlan \leftarrow neighbor$
16:                $bestLocalCost \leftarrow nCost$
17:                $bestLocalI \leftarrow nI$
18:    if $bestCost < bestLocalCost$ then
19:        $bestPlan \leftarrow bestLocalPlan$
20:        $bestCost \leftarrow bestLocalCost$
21:        $uBP \leftarrow 0$
22:    else
23:        $uBP \leftarrow uBP + 1$
24:    if $maxUnchangedBestPlan \leq uBP$ then
25:        $nextPlan, nextI \leftarrow$ DIVERSITYMECHANISM()
26:    else
27:        $nextPlan \leftarrow bestLocalPlan$
28:        $nextI \leftarrow bestLocalI$
29:    EXPLORENEIGHBOURHOOD($datasets[], t, nextPlan, nextI$)
```

and the set of interventions that are possible for changing the output dataset. On lines 3-4, the bias and the cost are computed for the resulting dataset. The bias computed by COMPUTE-BIAS is based on a chosen fairness metric as explained in Section V-A. The cost computed by COMPUTECOST is done by determining the ratio of null values in the output dataset. Then, on line 5, the evaluated plan is marked as *tabu* so that the system does not evaluate it again in subsequent iterations. Also, on line 6, the number of evaluated plans is increased by 1. This is relevant for checking the stopping criteria which has an upper limit for the number of evaluated plans.

EXPLORENEIGHBOURHOOD. Algorithm 3 shows the recursive method that explores the neighborhood of a *best local plan* at a given point. The *best local plan* represents the plan from a given neighbourhood that is the most cost-effective. The cost is computed through the objective function explained in Section V-C. On lines 2-3, the system checks if the stopping criterion is met. In our case, the search stops if the maximum number of evaluated plans is reached. On lines 4-6, the best local plan variables are initialized before the neighborhood exploration starts. On line 7, through GENNEIGHBOURS, the neighbour plans are generated as follows: each neighbor plan contains all interventions from the currently explored plan ($explPlan$) together with only 1 intervention from the proposed set of interventions I. The number of $neighbours$ will be equal to the number of interventions in I. On lines 8-17, each neighbour plan is evaluated and, as each is evaluated, the *best local plan* is updated with the most cost-effective plan. Before evaluating a plan, on lines 9-10, the system checks again if the stopping criterion is met as the upper limit of evaluated plans can be reached during the exploration of the neighbourhood. On line 18, the system checks if the *best overall plan* needs to be updated with the *best local plan*, i.e., if the *best overall plan* has a higher cost than the *best local plan*, then it is updated (lines 19-20). Also, on line 21, the uBP variable is reset to 0 as the best overall plan has been changed. Otherwise, if the *best overall plan* remains unchanged, then uBP is incremented. uBP variable keeps track of how many times a neighborhood has been explored without changing the *best overall plan*. On line 24, it is checked if uBP exceeds the maximum number of neighborhoods that have been explored without changing the best plan. If this upper limit has been reached, then the system runs the diversity mechanism on line 25. The diversity mechanism will yield the next plan to be explored (paired with its corresponding interventions), regardless of the fact that the plan is in the *tabu* list. If this upper limit has not been reached, then the search proceeds by exploring the neighborhood of the *best local plan* (line 29). The EXPLORENEIGHBOURHOOD method is recursively called until the stopping criteria is met.

VI. EXPERIMENTS

In this section we evaluate the performance of the wrangling pipeline w.r.t fairness using two types of experiments: Section VI-A shows how fairness can differ with varying number of sources, and Section VI-B explores the effect of the amount of searching for interventions on fairness and completeness.

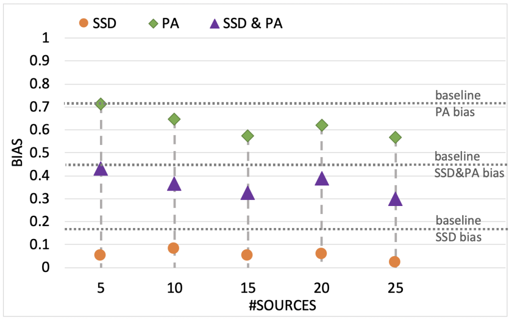
Input. *Sources.* For building the sources in our scenarios we used the Census Adult dataset [7]. We chose this dataset as it is known to be biased on both types of fairness metrics that we consider. The sensitive attribute is *gender* as it contains ≈66% *male* and only ≈33% *female* values. The potential proxy attribute is considered the *relationship* attribute as ≈41% of the relationship values can determine the gender value, e.g., the tuples contain pairs such as (*Husband*, *male*), (*Wife*, *female*).

The starting point is a single table dataset, but to experiment with data wrangling, several tables are needed. So, for the purposes of experimentation, the dataset was divided into smaller tables by using horizontal and vertical partitions of random sizes that, if correctly assembled back, can reconstruct the initial dataset. Besides the tables resulting from the partitioning process, we created a subset of tables where we duplicated tables where *relationship* attribute appeared, and we replaced the *Husband* and *Wife* values with *Spouse*, aiming to weaken the dependency between the *gender* and *relationship*. This provides alternative ways of populating the target with data for each individual, some of which are fairer than others.

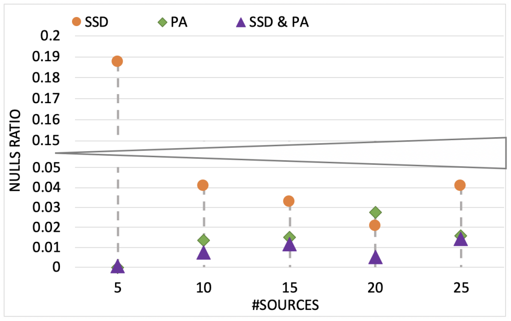
Target schema. The target schema is that of the initial Adult dataset [7], i.e., containing all 14 attributes.

Objective function. As explained in Section V-C, we use an objective function (shown in Eq. 4) for the trade off between bias and completeness. For these experiments we used weight of 0.7 for the biases component and 0.3 for the quality cost.

Evaluation metrics. We report the results in terms of *bias* and *quality* (*nulls ratio*, in our case) for three different cases: considering the bias coming only from *sample size disparity* (*SSD*), considering bias coming only from *proxy attributes*



(a) Fairness bias



(b) Cost on dataset quality - nulls ratio

Fig. 2. Performance on scenarios of varying numbers of sources

(*PA*), and bias coming from both types of strategies (*SSD & PA*). In the latter case, we report the weighted sum over the two measured biases where the weights are 0.35 each, thus, each type of bias has equal importance in the objective function. For comparison, we consider as *baseline* cases for bias the biases computed on the datasets with no interventions.

Experimental setup. The experiments were run over an Intel Core i5 with 2×2.7 GHz, and 8 GB RAM. We report the *bias* and *cost* results over the average of 5 runs for each scenario.

A. Fairness Bias vs. Number of Sources

This experiment shows how the number of data sources can affect the final outcome of the data wrangling process w.r.t. the chosen fairness metric(s). The number of sources can influence the output dataset as the more sources in the search space, the more options the data wrangling pipeline can explore for different, less biased outcomes. However, having many sources can come with the challenge of having a prohibitively large space to explore, thus, the likelihood of finding the optimal plan can decrease without a high limit of evaluated plans.

For this experiment, we used a set of 5 scenarios where we increased the number of sources from 5 to 25 (all stemming from the same Adult dataset). All sources have at least one join opportunity with the other sources such that the initial dataset can be recreated through at least one way. The upper limit for the number of evaluated plans was kept fixed at 100.

Results. The results are shown in Figures 2(a) and 2(b) for each scenario for *bias* and for *ratio of nulls*, respectively.

Bias. Figure 2(a) shows the results on bias of the output datasets. It can be observed that for all three cases, i.e., *SSD*, *PA* and *SSD & PA*, the bias values tend to decrease from the small scenario (5 sources) to the large one (25 sources). This is because the data wrangling process had multiple options for adding different data sources to the output, thus, obtaining less biased datasets. Also, it can be observed that the set upper limit of evaluated plans is high enough so that a way is found to merge the sources in a less biased way for all scenarios. This is observed by comparing each bias value with its corresponding baseline bias value, e.g., for the 5 scenario case, the *SSD* bias value is situated below the *SSD baseline*.

For the *PA* values, although almost all are below the *PA baseline*, it can be seen that all are above 0.5, which can be considered a rather high dependency outcome. This is due to the fact that in all scenarios, the *PA* bias was computed based on approximate functional dependencies with a coefficient higher than 0.5 between the sensitive attribute and *all* other attributes. Although it was expected that only *relationship* would be a considered *proxy* attribute, it resulted that, although less apparent, other attributes have high dependencies with the sensitive attribute as well, leading to a high *PA* bias.

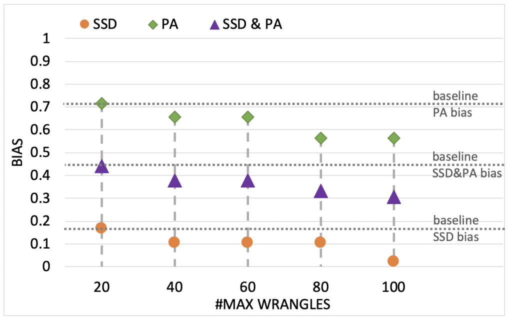
Quality. In our experiments, we measure the quality of the output dataset as the ratio of nulls. Figure 2(b) shows the result on the ratio of nulls for each of the 5 scenarios, where the corresponding biases are the ones in Figure 2(a). An interesting thing to observe here is the result on the scenario with 5 sources. Here, although its *SSD* bias had a lower value compared to the scenario with 10 sources, this has come with a trade off in quality, as the ratio of nulls is ≈4 times higher compared to that of the 10-source scenario. This happened because the system did not have many options for intervention plans to choose from so it picked the most suitable outcome out of the available plans. A similar behaviour can be observed for the other results, i.e., the higher the nulls ratio, the lower the corresponding bias (depicted in Figure 2(a)).

B. Fairness Bias vs. Amount of Explored Search Space

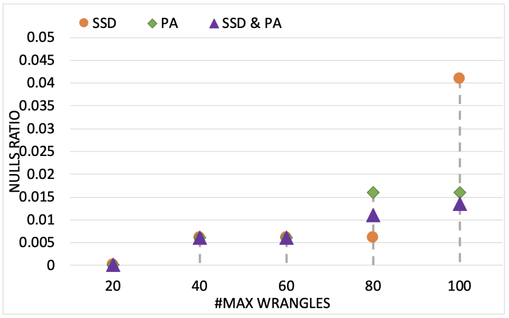
This experiment shows how the set upper limit of evaluated number of plans can affect the final outcome of the data wrangling process w.r.t. the chosen fairness metric(s). If the upper limit is too low for the search space, then reaching an optimal solution, i.e., a suitable intervention plan in terms of both bias and quality, is less likely. For this experiment we used the 25-source scenario from Section VI-A and we varied the upper limit for evaluated plans from 20 to 100.

Results. The results are shown in Figures 3(a) and 3(b) for each scenario for *bias* and for *ratio of nulls*, respectively.

Bias. In Figure 3(a) one can observe that, in the first scenario, the wrangling process does not succeed at improving the bias for any of the fairness metrics as they are all on their corresponding *baseline*. However, in Figure 3(b), it can be seen that for the same case, the ratio of nulls is 0, meaning that, although the dataset is biased, the 25 sources were correlated as expected. The trend line is oriented downwards in all cases, *SSD*, *PA* and *SSD &PA*, meaning that the bias decreases as the number of evaluated plans increases. This is due to the fact



(a) Fairness bias



(b) Cost on dataset quality - nulls ratio

Fig. 3. Performance on scenarios of varying the upper limit of explored plans

that more parts of the search space are explored, thus, the system can search more for a plan with a smaller bias.

Quality. Similarly to Section VI-A, Figure 3(b) shows that, as the bias starts decreasing, the ratio of nulls starts increasing. This is partly a characteristic of these scenarios, but the interventions can involve removing problematic matches and joins, changes that will tend to reduce completeness. Despite this, it is generally possible that both bias and nulls ratio could decrease through the same interventions.

VII. Conclusions

In this section, we revisit the claimed contributions in the introduction, and elaborate how these were realised in the paper.

1) *A characterization of problems that may arise during data wrangling that may lead to bias in downstream analyses*. We have identified *sample size disparity* and *proxy attributes* as issues that can arise and be identified during data wrangling.
2) *A proposal for how to detect such problems and respond to them in a system that automates the generation of data wrangling pipelines*. We described how to detect *sample size disparity* and *proxy attributes* in mapping results, and provided techniques to identify interventions that may *(i)* reduce the occurrence of these features; and *(ii)* enable the identification of more suitable ways of wrangling the data.
3) *A search algorithm that explores alternative ways of wrangling the data with a view to producing fair results from data wrangling*. Given a space of possible actions, we have described a search process that investigates how combinations of these actions can be used together to increase the predicted fairness of the wrangling result.
4) *An evaluation of (3) for a benchmark data set that has been used to study fairness interventions*. We have applied the technique to a real-world data set, which has shown how the levels of *sample size disparity* and level of dependency of a sensitive attribute on *proxy attributes* can be reduced.

Acknowledgments. We are pleased to acknowledge the support of the UK Engineering and Physical Sciences Research Council, through the VADA Programme Grant (EP/M025268/1).

References

[1] S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. Last accessed 15 June 2020.
[2] J. Bleiholder, S. Szott, M. Herschel, F. Kaufer, and F. Naumann. Subsumption and complementation as data fusion operators. In *EDBT '10*, pages 513–524, 2010.
[3] T. Calders, F. Kamiran, and M. Pechenizkiy. Building classifiers with independency constraints. pages 13–18, 12 2009.
[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, June 2002.
[5] N. V. Chawla, L. O. Hall, and A. Joshi. Wrapper-based computation and evaluation of sampling methods for imbalanced datasets. In *UBDM*, page 24–33, New York, NY, USA, 2005. ACM.
[6] B. d'Alessandro, C. O'Neil, and T. LaGatta. Conscientious classification: A data scientist's guide to discrimination-aware classification. *Big Data*, 5:120–134, 06 2017.
[7] D. Dua and C. Graff. UCI machine learning repository, 2017.
[8] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. Certifying and removing disparate impact. In *ACM SIGKDD*, KDD '15, page 259–268, New York, NY, USA, 2015. ACM.
[9] S. A. Friedler, C. Scheidegger, S. Venkatasubramanian, S. Choudhary, E. P. Hamilton, and D. Roth. A comparative study of fairness-enhancing interventions in machine learning. In *FAT**, pages 329–338. ACM, 2019.
[10] T. Furche, G. Gottlob, L. Libkin, G. Orsi, and N. W. Paton. Data wrangling for big data: Challenges and opportunities. In *EDBT*, pages 473–478, 2016.
[11] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. In *NIPS*, page 3323–3331, Red Hook, NY, USA, 2016.
[12] F. Kamiran and T. Calders. Classifying without discriminating. *Computer, Control and Communication*, pages 1–6, 2009.
[13] F. Kamiran and T. Calders. Data preprocessing techniques for classification without discrimination. *KAIS*, 33:1–33, 2011.
[14] F. Kamiran, A. Karim, and X. Zhang. Decision theory for discrimination-aware classification. In *ICDM*, page 924–929, USA, 2012.
[15] T. Kamishima, S. Akaho, H. Asoh, and J. Sakuma. Fairness-aware classifier with prejudice remover regularizer. pages 35–50, 09 2012.
[16] N. Kilbertus, M. Rojas-Carulla, G. Parascandolo, M. Hardt, D. Janzing, and B. Schölkopf. Avoiding discrimination through causal reasoning. In *NIPS*, page 656–666, Red Hook, NY, USA, 2017.
[17] S. Köknar-Tezel and L. J. Latecki. Improving svm classification on imbalanced time series data sets with ghost points. *Knowl. Inf. Syst.*, 28(1):1–23, July 2011.
[18] N. Konstantinou, E. Abel, L. Bellomarini, A. Bogatu, C. Civili, E. Irfanie, M. Koehler, L. Mazilu, E. Sallinger, A. Fernandes, G. Gottlob, J. Keane, and N. Paton. Vada: An architecture for end user informed data preparation. *Journal of Big Data*, 6:74, 2019.
[19] S. Kruse and F. Naumann. Efficient discovery of approximate dependencies. *VLDB*, 11(7):759–772, 2018.
[20] L. Mazilu, N. W. Paton, A. A. Fernandes, and M. Koehler. Dynamap: Schema mapping generation in the wild. In *SSDBM*, page 37–48, 2019.
[21] V. Zelaya, P. Missier, and D. Prangle. Parametrised data sampling for fairness optimisation. In *KDD XAI*, 2019.
[22] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *ICML*, ICML'13, page III–325–III–333, 2013.
[23] B. H. Zhang, B. Lemoine, and M. Mitchell. Mitigating unwanted biases with adversarial learning. In *AAAI*, page 335–340, NY, USA, 2018.