

MithraLabel: Flexible Dataset Nutritional Labels for Responsible Data Science*

Chenkai Sun [†], Abolfazl Asudeh [‡], H. V. Jagadish [†], Bill Howe ^{‡†}, Julia Stoyanovich ^{††}

[†]University of Michigan; [‡]University of Illinois at Chicago; ^{‡†}University of Washington; ^{††}New York University

ABSTRACT

Using inappropriate datasets for data science tasks can be harmful, especially for applications that impact humans. Targeting data ethics, we demonstrate MithraLabel, a system for generating task-specific information about a dataset, in the form of a set of visual widgets, as a flexible "nutritional label" that provides a user with information to determine the fitness of the dataset for the task at hand.

KEYWORDS

Data Ethics; Machine Bias; Fairness; Accountability; Transparency

ACM Reference Format:

Chenkai Sun [†], Abolfazl Asudeh [‡], H. V. Jagadish [†], Bill Howe ^{‡†}, Julia Stoyanovich ^{††}. 2019. MithraLabel: Flexible Dataset Nutritional Labels for Responsible Data Science. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19)*, November 3–7, 2019, Beijing, China. ACM, New York, NY, USA, 4 pages. <https://doi.org/10.1145/3357384.3357853>

1 INTRODUCTION

Given the proliferation of available datasets, data scientists and machine learning experts often struggle to choose among alternative datasets for a given task. A data scientist must evaluate each candidate dataset to determine its fitness for use. For example, a linear regression task implies normality assumptions about the measurement error, and any prediction task assumes that the sample is representative of the overall population. These assumptions can be evaluated with statistical tests, but this level of rigor is rarely applied in practice, with potentially serious consequences.

These cases, unfortunately, are not rare. An example is Google's *gorilla incident* [1]. An early image tagging algorithm released by Google mistook people with dark skin for animals, incurring significant *representational harms* [2] for a legally-protected demographic group. The cause was the scarcity of dark-skinned faces in the dataset used for training. In other words, the problem was caused by the training dataset's lack of fitness for the given task.

*This work was supported in part by NSF Grants No. 1926250, 1741022, 1740996, and Microsoft.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM '19, November 3–7, 2019, Beijing, China

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6976-3/19/11...\$15.00

<https://doi.org/10.1145/3357384.3357853>

The effects of using inappropriate training dataset may become tragic when it comes to applications that cause *allocative harms* [2] in domains such as employment, health care, and criminal justice. For example, judges in many jurisdictions consider risk scores assigned to individuals by risk assessment software, based on their criminal record and their background, as guidance when deciding on bail and sentencing. ProPublica [3] showed that these scores exhibit systemic racial bias — black defendants were twice as likely as whites to be incorrectly judged high-risk for recidivism yet not re-offend (false positives), while white defendants were twice as likely to be judged low-risk compared to blacks, yet go on to re-offend (false negatives). These racial disparities are in part due to historical discrimination in the criminal justice system encoded in the training and validation datasets. Importantly, because of concerns of disparate treatment — explicitly treating individuals differently based on legally-protected characteristics such as race, disability status or gender — race is not one of the inputs to the software. However, this *blinding* does not help counteract racial disparities in prediction, due to correlations between race and other attributes in the data.

Considering *Data Ethics and Responsible Data Science* as our main target in our system, ideally, we would like to create a system that provides immediate *information* for a data scientist about the fitness of a dataset to a task she has in mind. Such information includes

- *Representativeness of minorities*: A dataset is usually required to be a representative sample of some population distribution. But representativeness is not enough: to inform useful models, datasets must include sufficient examples from minority classes, or it may not learn the behaviour of the those groups leading to inequities such as *disparate predictive accuracy* [4], as occurred in Google's gorilla incident.
- *Bias*: Bias in the input data is a major reason for machine bias and incidents such as [3]. Fortunately bias and its related problems have recently been a focus of data ethics community [5–9] and different definitions such as statistical bias and societal bias has been studied [10].
- *Correctness*: Do all records in the dataset satisfy the conditions required by the specified method? For example, missing training labels may generate unexpected errors, or some records may not be of people.

While not limiting the scope of *MithraLabel*, in this demo we focus on providing information for the above properties.

1.1 Related work

Datasets may come equipped with basic metadata or *data profiling* [11] information: the number of records, the arity of the schema, the cardinality of an attribute, etc. However, fitness for use for a

specific task requires additional statistical tests to be run and presented for interpretation, as discussed above; these tests are outside the scope of typical task-agnostic profiling tools.

Orthogonal to data profiling, *data exploration* techniques [12] provide tools for the user to explore the data and find interesting patterns. Unlike data profiling, these techniques provide dynamic content; however, they are not task-aware and put the data exploration burden to the user. Inspired by nutritional labels found on packaged foods, RankingFacts [13] proposes to use the metaphor of a nutritional label to make data and models interpretable. In particular, it defines a nutritional label as a set of automatically constructed visual *widgets*, and shows how this kind of a standardized representation can convey information for the output of a ranking process.¹ A follow-on work [14] proposes a (manually constructed) nutritional label for a dataset. Other follow-on works propose similar, manually constructed, data sheets [15] or model cards [16].

1.2 Contributions

Extending the idea in [13], we propose a nutritional label as a set of visual widgets over a dataset. While [13] focuses on the *output* of rankers, here our focus is to generate labels for the datasets as the *input* to different tasks. Our proposal is different from data profiling in that it is *task-specific*, and different from data exploration in that there is a specific goal: expose properties that demonstrate fitness for a given task. Considering the dataset as a collection of items over a set of attributes, each widget provides specific information (such as functional dependencies) about the whole dataset or some selected part of it. For example, if a data scientist is considering the use of a number-of-prior-arrests attribute to predict likelihood of recidivism, she should know that the number of prior arrests is highly correlated with the likelihood of re-offending, but it introduces *bias* as the number of prior arrests is much higher for African Americans than for other races due to policing practices and segregation effects in poor neighborhoods. Some widgets that can appear in the nutritional label for this (prior arrests) attribute are: count of missing values, correlation (association between ordinal variables) with the predicted attribute or a protected attribute, and the distribution of values.

One could ask: why not providing all possible information about a dataset? That, in fact, would be overwhelming for the users as their short attention spans require to see the relevant widgets corresponding to their tasks. Hence, we propose the “*widget selection*” problem to identify an optimal set of widgets for a given task, within a limited space budget. Given a specific task, each widget provides some information (profit) and takes some space in the label (cost). In the design of our system, we put an emphasis on flexibility and automatic learning of profits. Furthermore, we make the interface customizable: users can select the widgets manually, and can also decide the number of widgets to be included in the label. They can also customize the automatically generated label by replacing some of the widgets. We shall elaborate on these in § 2.2.

¹The idea of a nutritional label for data and models, and its instantiation on score-based rankers, was first proposed in 2016: <http://freedom-to-tinker.com/2016/08/05/revealing-algorithmic-rankers>.

MithraLabel System

The screenshot shows the MithraLabel System interface. It has two main sections: 'Upload your dataset' and 'Choose a sample dataset'. Under 'Upload your dataset', there is a 'Select .csv file' button and an 'Upload' button. Under 'Choose a sample dataset', there is a dropdown menu showing 'RecidivismData_Original.csv' and a 'Confirm' button. Below these is the 'Specify a task' section with three radio buttons: 'Classification', 'Ranking' (which is selected), and 'Clustering ...'. The 'Selections' section follows, with two radio buttons: 'Single Column Analysis' and 'Multi-Column Analysis' (which is selected). Below this is a 'Pick attributes' section with a 'warning' icon and a list of attributes: 'Violence_score', 'decile_score', 'first_name', 'age', 'marriage_status', 'c_charge_degree', and 'event'. There is a 'clear all' button and a dropdown arrow. Below that is a 'Pick protected/label attributes' section with 'race' and 'sex' selected, and a 'clear all' button and a dropdown arrow. Then is a 'Pick widgets yourself' section with two widgets: 'Maximal Uncovered Patterns' and 'Functional Dependencies', and a 'clear all' button and a dropdown arrow. Finally, there is a 'Slice the dataset by value range' section with 'age' selected, a range input showing 'age: [20 , 70]', and a 'clear all' button and a dropdown arrow.

Figure 1: Components of the MithraLabel system

2 SYSTEM OVERVIEW

In this section, first in § 2.1, we discuss our current support widgets and tasks. Next in § 2.2, we elaborate on the technical details of fitness for use, and finally provide the system architecture in § 2.3.

2.1 Widgets and Tasks



Our proposal in this paper is to design *MithraLabel* as tool for supporting responsible data science by generating nutritional labels that show information about the fitness of a dataset for a specific task. Our system relies on a universe of widgets and a set of task categories. As mentioned in § 1, the focus of this demo is to include widgets that provide information about (i) *representativeness of minorities*, (ii) *bias*, and (iii) *correctness* properties. Please note that the scope of MithraLabel is not limited to these and arbitrary widgets and tasks can be added to it. In § 2.2, we will discuss how this has been considered in the design of the system.

In our selection of an initial set of tasks and widgets, we first decided on a set of tasks that we believed are more important, at least for human-sensitive applications. For instance, we identified evaluating individuals and ranking as an important step in decision making that can impact human lives and society and is highly criticized [3, 17]. Besides (a) ranking, we also identified (b) label assignment and classification, (c) prediction, and (d) clustering.

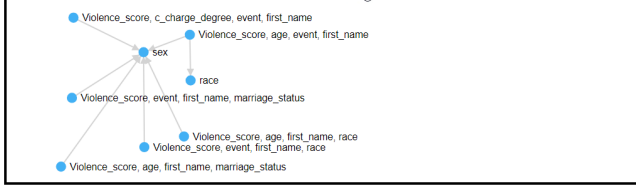
In process of identifying the widgets, we focused on our pool of tasks and the aforementioned properties (i), (ii), and (iii). For example the widgets (1) *correlation* and (2) *functional dependencies* with protected attributes and output variables, and (3) *association rules* to capture *bias*. For representativeness of minority groups, we consider (4) *maximal uncovered patterns* (MUPs), the results of our recent paper [18] for identifying the regions in the data cube

[Data Overview](#) [Functional Dependencies](#) [Maximal Uncovered Patterns](#)

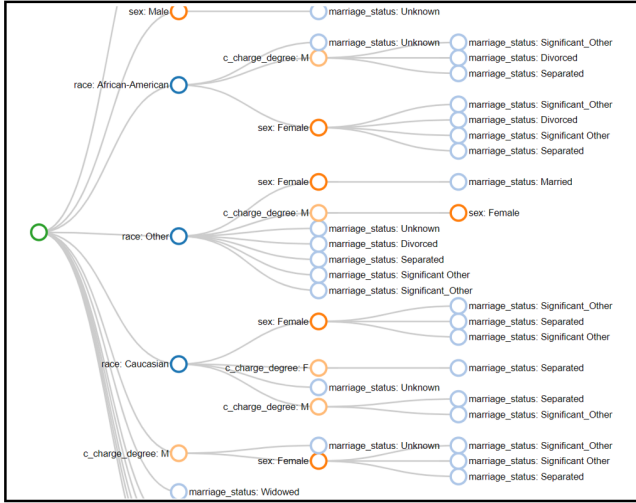
Data Overview (Please wait while the widgets are rendering)

Attribute Name	Histogram	Max	Min	Mean	Null Entries	Unique Entries
Recidivism_score		1.69	-3	-0.69	0	33
Violence_score		0.93	-4.63	-2.37	0	39

Functional Dependencies



Maximal Uncovered Patterns



Generate More Labels

Figure 2: Label page

for which there are not enough representatives. In addition, we also consider the following widgets: (5) *general information*: the general overview containing, for example, the number of tuples, number of attributes, nature of attributes, and their cardinalities, (6) *value distributions*: the value distribution and statistical information about each column, (7) *summary sketch*: (inspired from [19]) a visual summary in the form of a compact table of the representative tuples/attributes of the dataset, (8) *missing values*: the overall percentage of missing values, as well as attributes and tuples with high ratio of missing values, (9) *outliers*, (10) *demographic parity* [5]: the attributes with maximum/minimum demographic parity (with regard to the protected attributes) in their top values, and (11) *diversity* [20]: the most diverse attributes on demographic groups [13]. Among these, the widgets 4 and 11 provide information about property (i), while widgets 1, 2, 3, 6, 10, and 11 support (ii) and widgets 5, 6, 7, 8, and 9 satisfy property (iii). Figure 2 illustrates an example label that contains the widgets 2, 4, and 5.

2.2 Flexible Labels and Adaptive Fitness

The system allows the user to choose a subset of widgets to be included in the nutritional label, or let the system select widgets automatically. Considering the “information” a widget adds to a nutritional label as its *profit* and the amount of space it takes as its *cost*, we define the “*widget selection*” problem (WSP) that identifies the optimal nutritional label as a set of widgets that their total cost is less than a budget and the profit of the set is maximized. Due to the mutual information [21] between the widget, the profit of a widgets in a label depends on the other widgets that has been selected. Using the reduction from the Quadratic Knapsack problem (QKP) [22], we prove the following theorems:

THEOREM 1. *WSP is NP-complete.*

THEOREM 2. *There is no P-time approximation algorithm with a fixed approximation ratio for WSP unless P=NP.*

Since further details about this is out of the scope of this demo, we leave this research problem, as well as the proof of theorems, to our long version paper. In this demo, we consider a simplified version of the problem, in which the cost of widgets are equal and the widgets are considered independent. This transforms the problem to *selecting the top-k widgets* (for a user specified value k) for a given task.

Still a challenge is left: WSP assumes the existence of a *profit table* relating the profit of a widget. Given that this does not hold in practice, we exploit users’ feedback for learning this table: the log of previous queries can be used to infer the profit. This idea hinges on the observations that the bulk of data scientists are expected to behave rationally. Thus, widgets that are more informative for a specific task are expected to be selected more often for that task. In the profit table every row corresponds to a widget and every column corresponds to a task. Initially, all cells are zero². We consider a value of 1 for every manually constructed (or adjusted) nutritional label and evenly divide this value to the profit of its widgets. That is, if there are k widgets in a label, a value of $1/k$ is added to the profit of each of those for the given task in the profit table. The reason behind this is that the more the widgets in a label, the less the importance of each one.

The automatic widget selection picks the top- k widgets with maximum profit for the given task, based on the profit table. We expect the user to adjust the final label by replacing irrelevant widgets, in which cases we adjust the profit table accordingly.

In addition to the nutritional label for the complete dataset, MithraLabel can generate focused information about different parts of the datasets, as needed. For example, consider a user who wants to use a dataset for a task such as prediction. Using our system, she first reviews overview information about the dataset and its fitness for her task. She may then decide to include an attribute in the predictive model. Using MithraLabel, she can construct a nutritional label that shows the fitness of this attribute for the prediction, and includes widgets such as missing values, value distribution and correlation with the label attribute. In addition, the user may include widgets that highlight a specific record of importance. For example, a nutritional label for a tuple t related to a prediction task may include the density in the neighborhood of t , if the item is an

²We use the crowd of graduate students in our lab for the cold-start issue.

outlier, or an extremal point. Similarly, the user can limit the scope of the label to a subset of attributes and items.

Figure 1 shows an example in which the user has selected the “COMPAS” recidivism dataset³, while specifying *ranking* as the desired task. Limiting the Nutritional Label to a subset of attributes (violence_score, etc.), race and sex are selected as protected attributes. A filtering condition has been applied to limit the label to individuals with ages in range [20,70], while the widgets are selected manually. The result label is shown in Figure 2.

As the final note in this subsection, we would like to highlight that using the idea of automatic learning of profits, our system is a flexible framework for adding new widgets and tasks. Adding new tasks are as simple as adding new columns to the profit table and initiating its column values with default values. Adding new widgets, however, requires the implementation of an oracle that, given the data, generates the corresponding widget.

2.3 Implementation Details

Our system is implemented as a web application. Clients use the Front-end to identify a dataset, a task, and the label details. The request will then be passed to the back-end where the nutritional label is constructed. The back-end is implemented in Python 3.5.2. We used the Flask framework to build web services and Pandas library to perform actions on the dataset. For front-end, in addition to HTML and CSS, we use React 16.0.0 to parse the information. A back-end DB is used to store the parameters/variables, query logs, and the profit table. As an implementation note: we offer sampling for efficient label construction for large datasets.

3 DEMONSTRATION PLAN

MithraLabel is online: <http://mithra.eecs.umich.edu/demo/label/>. The demonstration includes (1) Recidivism Dataset: a dataset collected and published by ProPublica [3] that includes the background records of 6.8K criminals and has been widely criticized for being racist, (2) CSMetrics Dataset: collected from CSMetrics⁴, a website for ranking CS institutes based on their measured_citation and predicted_citation.

Input: During the demonstration, the users can upload/select a dataset and select a task (Figure 1). The input section is shown in Figure 1. It allows the client to specify the dataset, a task, and extra information such as attributes of interest. We use tooltips/warnings to help the client understand what each subsection does:

- *Single column analysis and Multi-column analysis*: besides multi-column analysis, we consider single column analysis as a special case that provides insights tailored about a single attribute and targets applications such as feature engineering.
- *label and protected attributes*: specifies the label and sensitive attributes such as race and gender.
- *Pick widgets yourself*: gives the user the ability to choose the widgets manually, instead of using our widget selector.
- *Slice the dataset by value range*: allows the user to filter the data and create the label for the specified slice.

As mentioned in § 2.2, Figure 1 showcases creating a nutritional label for the COMPAS dataset and ranking as the underlying task, while limiting the label to a subset of attributes and the entries with

ages between 20 and 70. Finally, the user has the option to choose our widget selector or to manually choose the label’s widgets.

Output: This section provides the label, in the form of a set of widgets, rendered vertically (Figure 2). The user can adjust the label by adding or removing widgets. During the demonstration, the presenter will guide users in exploring and understanding each of the selected widgets. For example, exploring the MUPs widget in Figure 2, we will explain which minority subgroups are not covered in the COMPASS dataset and discuss its potential harms.

4 FINAL REMARKS

In this paper we proposed MithraLabel, a tool for assisting responsible data scientists to conduct data science tasks using available datasets. MithraLabel provides flexible nutritional labels for datasets in the form of a set of visual information units, known as widgets. While the current focus of MithraLabel is on properties correctness, bias, and representativeness of minorities, the system is not limited to these and additional widgets and tasks can gradually be added to it. In § 2.2, we introduced the widget selection problem for identifying the optimal nutritional label for a given task. Studying this problem is part of our ongoing project for the long version paper.

REFERENCES

- [1] M. Mulshine. A major flaw in google’s algorithm allegedly tagged two black people’s faces with the word ‘gorillas’. In *BusinessInsider*, 2015.
- [2] S. Barocas, K. Crawford, A. Shapiro, and H. Wallach. The problem with bias: Allocative versus representational harms in machine learning. In *SIGCIS*, 2017.
- [3] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias: Risk assessments in criminal sentencing. *ProPublica*, 2016.
- [4] I. Chen, F. D. Johansson, and D. Sontag. Why is my classifier discriminatory? In *NeurIPS*, 2018.
- [5] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *ITCS*, 2012.
- [6] J. Kleinberg, J. Ludwig, S. Mullainathan, and A. Rambachan. Algorithmic fairness. In *AEA Papers and Proceedings*, 2018.
- [7] 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**, 2019.
- [8] A. Asudeh, H. Jagadish, J. Stoyanovich, and G. Das. Designing fair ranking schemes. In *SIGMOD*, 2019.
- [9] B. Salimi, L. Rodriguez, B. Howe, and D. Suciu. Capuchin: Causal database repair for algorithmic fairness. In *SIGMOD*, 2019.
- [10] A. Narayanan. Translation tutorial: 21 fairness definitions and their politics. In *FAT**, 2018.
- [11] Z. Abedjan, L. Golab, F. Naumann, and T. Papenbrock. Data profiling. *Synthesis Lectures on Data Management*, 10(4):1–154, 2018.
- [12] S. Idreos, O. Papaemmanouil, and S. Chaudhuri. Overview of data exploration techniques. In *SIGMOD*, pages 277–281, 2015.
- [13] K. Yang, J. Stoyanovich, A. Asudeh, B. Howe, H. V. Jagadish, and G. Miklau. A nutritional label for rankings. In *SIGMOD*, 2018.
- [14] S. Holland, A. Hosny, S. Newman, J. Joseph, and K. Chmielinski. The dataset nutrition label: A framework to drive higher data quality standards. *CoRR abs/1805.03677*, 2018.
- [15] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. M. Wallach, H. D. III, and K. Crawford. Datasheets for datasets. *CoRR*, abs/1803.09010, 2018.
- [16] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L. Vasserman, B. Hutchinson, E. Spitzer, I. D. Ravi, and T. Gebru. Model cards for model reporting. In *FAT**, 2019.
- [17] A. Asudeh, H. Jagadish, G. Miklau, and J. Stoyanovich. On obtaining stable rankings. *PVLDB*, 12(3):237–250, 2018.
- [18] A. Asudeh, Z. Jin, and H. V. Jagadish. Assessing and remedying coverage for a given dataset. In *ICDE*, 2019.
- [19] N. Yan, S. Hasani, A. Asudeh, and C. Li. Generating preview tables for entity graphs. In *SIGMOD*, 2016.
- [20] M. Drosou, H. Jagadish, E. Pitoura, and J. Stoyanovich. Diversity in big data: A review. *Big data*, 5(2):73–84, 2017.
- [21] S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.
- [22] C. Witzgall. Mathematical methods of site selection for electronic message systems (ems). *NASA STI/Recon Tech. Report*, 1975.

³propubli.ca.org/dataset/compas-recidivism-risk-score-data-and-analysis

⁴www.csmetrics.org