Metadata Generation and Nutrition Label

Tianyu Wang
University of Rochester
twang83@ur.rochester.edu

Moshiul Azam
University of Rochester
mazam@ur.rochester.edu

1. Introduction

Data has become part of our life. Many of these data impact us directly. Using inappropriate data can be harmful. Sometimes data scientists and machine learning engineers struggle to find the balance and choose the wrong data sets for the given tasks. To avoid these problems a system for generating task specific information 'nutritional label' provides users with useful information to determine the usefulness of the dataset. There are much more datasets on the Web than data warehouses, but most of them have no metadata and description about the dataset. Proper description and metadata can help, for example, machine learning researchers to choose unbiased data to train their model, the biased data definitely will cause low accuracy and robustness. Since it is commonplace to have a learning algorithm trained based on some dataset, a data scientist may be able to specify requirements such as representation and coverage. However, more often than not, analyses are done with data that has been acquired independently. More recently, it has been recognized that it is not enough for the training data to be representative: it must have enough examples from less popular "categories". The best known story underlining the importance of this inclusion is the case of "google gorilla" [1]. An early image recognition algorithm released by Google had not been trained on enough dark-skinned faces. When presented with an image of a dark African American, the algorithm labeled her as a "gorilla". While Google very quickly patched the software as soon as the story broke, the question is what it could have done beforehand to avoid such a mistake in the first place. This is not a unique case. The solution of Google is to ban 'gorilla' from their library. This is not a good solution as we know, this ban also limits the performance of their system.

The problem becomes critical when it comes to data-driven algorithmic decision making. For example, judges, probation and parole officers are increasingly using algorithms to assess a criminal defendant's likelihood to re-offend [5]. Image such a tool can provide insightful signals for the judge and have the potential to make society safer, this means a wrong signal can have devastating effects on an individual's lives. Hence, it is critical to make sure that tool is trained on data that includes adequate representation of individuals similar to each criminal that will be scored by it, especially for those sensitive attributes, including gender, race, age, nationality, and so on. Whether there is enough collection of data about these sensitive attributes is critical to the learning algorithms.

2. Preliminaries

We follow the definition from Abolfazl [2]. We consider a dataset D with d low-dimensional categorical attributes, A = $A_1, A_2, ..., A_d$, where attributes are continuous valued or of high cardinalities. Y = $y_1, ..., y_{d'}$ is "label attribute". The cardinality of an attribute A_i is c_i.

Definition 1 (Pattern). A pattern P is a vector of size d, in which P[i] is either X (meaning that its value is unspecified) or is a value of attribute A_i. We name the elements with value X as non-deterministic and the others are deterministic.

$$M(t,P) = \begin{cases} \top, & \forall i \in [1,d] : P[i] = X \text{ or } P[i] = t[i] \\ \bot, & \text{otherwise} \end{cases}$$

Figure 1. Formal definition of Pattern.

An item t matches a pattern P (written as M(t,P) = T), if for all i for which P[i] is deterministic, t[i] is equal to P[i].

For example, consider the pattern P = 1X0X on four binary attributes $A_1 to A_4$. Hence, for example, t_1 = [1,1,0,0] and t_2 = [1,0,1,0] match P. On the other hand, t_3 = [1,0,1,0] does not match P.

Definition 2 (Coverage). Given a dataset over d attributes with cardinalities c = $c_1 ... c_d$, and a Pattern P based on c and d, the coverageof P is the number of items in D that match P.

Since our goal is to check whether there are enough tuples for each value combination of sensitive attributes in the dataset. Here we choose threshold as '¿25', Sudman [3] suggests that for each 'minor subgroup' a minimum of 20 to 50 samples in necessary.

Definition 3 (Covered/Uncovered Pattern). A pattern P is said to be covered in a dataset D if its coverage is greater than the threshold, otherwise, P is said to be uncovered.

Definition 4 (Parent/Child Pattern). A pattern P_1 is a parent of a pattern P_2 if P_1 can be obtained by replacing one of the deterministic elements in P_2 with X. We can equivalently say that P_2 is a child of pattern P_1.

Definition 5 (Maximal Uncovered Pattern (MUP)). Given a threshold t, a pattern P is a maximal uncovered, if coverage(P) $<$ t, while for any pattern $P^{'}$ of P, coverage($P^{'}$) $>=$ t.

3. Related Work

3.1. Sherlock: A Deep Learning Approach to Semantic Data Type Detection

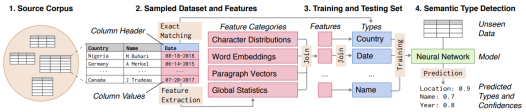

Figure 2. Data processing and analysis flow, starting from (1) a corpus of real-world datasets, proceeding to (2) feature extraction, (3) mapping extracted features to ground truth semantic types, and (4) model training and prediction.

Sherlock [4] is a multi-input neural network that can take large number of features and samples as input. First, it will check the existing header, if there are some exact matched column names with library, it will use this header, and its value to check. Otherwise, it will predict the header bsaed on the column values. Since there are a lot of data tables without headers in the open data lake. This system is so useful and critical to our current work. Sherlock has 0.89 on F1 score on their testing dataset, which is higher than Decision tree and Random forest. Also, Sherlock performs well on Grades, ISBN, Birth Data, Industry, and Affiliation attributes.

Method	F_1 Score	Runtime (s)	Size (Mb)
Machine Learning			
Sherlock	0.89	0.42 (±0.01)	6.2
Decision tree	0.76	0.26 (±0.01)	59.1
Random forest	0.84	0.26 (±0.01)	760.4
Matching-based			
Dictionary	0.16	0.01 (±0.03)	0.5
Regular expression	0.04	0.01 (±0.03)	0.01
Crowdsourced Annotations			
Consensus	0.32 (±0.02)	33.74 (±0.86)	–

Figure 3. Support-weighted F1 score, runtime at prediction, and size of Sherlock and four benchmarks. Data from Sherlock [4].

3.2. Assessing and Remedying Coverage for a Given Dataset

This paper shows us the idea of pattern, pattern graph, uncovered pattern, coverage, and MUP. It defines these core concepts, and it provides three kinds of algorithms to find MUP in the pattern graph, top-down algorithm, bottom-up algorithm, and deep-diver algorithm. Here we just focus on top-down algorithm.

Type	F_1 Score	Precision	Recall	Support
Top 5 Types				
Grades	0.991	0.989	0.994	1765
ISBN	0.986	0.981	0.992	1430
Birth Date	0.970	0.965	0.975	479
Industry	0.968	0.947	0.989	2958
Affiliation	0.961	0.966	0.956	1768
Bottom 5 Types				
Brand	0.685	0.760	0.623	574
Person	0.630	0.654	0.608	579
Director	0.537	0.700	0.436	225
Sales	0.514	0.568	0.469	322
Ranking	0.468	0.612	0.349	439

Figure 4. Top five and bottom five types by F1 score. Data from Sherlock [4].

3.3. MithraLabel: Flexible Dataset Nutritional Labels for Responsible Data Science

we propose a nutritional label as a set of visual widgets over a dataset. While 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.

In this paper we proposed Nutrition Label, 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. Using this program, user can tackle multiple aspects of any dataset like correctness, biases, representativeness.

From the Meta-data generated by our program , datascientist can tackle the correctness of the datasets, like how many missing values, distributions of attributes, distinctions. From correlations of attributes , user can assess a possible linear association between two continuous attributes.

From Functional dependencies, one can evaluate if dataset has any biases, if so what are those attributes which contributes to those biases , and user can then take actions ether collect more data or manipulate those dependent attributes in order to balance the overall dataset.The representativeness of any given dataset can be found from our association rules, which indicates subset of attributes that are uniquely determines other subset of attributes. By this association graph, user can determine the representativeness of

any individual attributes and collection of attributes.(How well any attribute represented in this dataset). Then user can add attributes that are not enough sampled or reduced other attributes which are over-sampled.This program provides the label, in the form of a set of widgets, rendered vertically. The user can adjust the label by adding or removing widgets.

4. Methods

We describe our workflow into two parts, first is metadata generation, second is nutrition label generation.

4.1. Metadata Generation

Metadata is used to provide a brief idea of the data table that can help data scientists have a basic understanding about the data table. This can help them choose the appropriate data tables to work with their learning algorithms.

First, we will check whether these exists header in the data table. If so, we will try to filter out some exact matches of sensitive attributes. If not, we will still use Sherlock system to predict a header based on the attribute values.

Once we filter out the sensitive attributes, we will determine the MUP and Coverage. Here we use a top-down algorithm as discussed in Abolfazl [2]. The value of P[i]

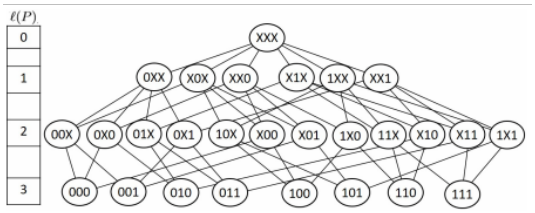

Figure 5. The pattern graph for three binary attributes. Graph from Abolfazl [2].

for a node in the pattern graph is either X or one of the values the corresponding attribute can take. Hence, the total number of nodes in a pattern graph defined over d attributes is $(p+1)^k$, where p is number of possible values, and k is number of attributes. Figure 2 contains $(2+1)^3 = 27$ nodes. Any pattern graph has only one node XX...X at level 0. In the level 1, the node is converted from root node on level 0 by specify a value to one of non-deterministic value X. We can iterate all nodes in the patern graph to find out the uncovered attributes and MUP. But we find, for example, in the adult census data table [6], there are four sensitive attributes, once we construct the corresponding pattern graph, there are about 800 nodes in the graph. If we iterate over all nodes, there are so many repeated iteration over the date table, which results in computation waste.

The idea of top-down algorithm is to prune some brunches and nodes based on monotonicity. We start from nodes in the level 1. If it is covered, we will check other nodes in the level 1, and push children of this node into stack. If we find this node is uncovered, we will prune all children of it, and check next node in the stack. This can help us prune a lot of brunch, especially when we can find MUPs on the top-part of the graph, but this also means it might have poor performance if most MUPs are located in the bottom part of the graph. Top0down algorithm is as same as iteration over all nodes, if there is no MUP, means all tuples are covered.

Algorithm 1 PATTERN-BREAKER
Input: Dataset $\mathcal{D}$ with d attributes having cardinalities c and threshold τ
Output: Maximal uncovered patterns $\mathcal{M}$

$Q = \{XX\cdots X\}$ // start from the root
$\mathcal{M} = \{\}$; $Q_p = \{\}$
for /* each level of the graph */ $l = 0$ to d **do**
  if Q is empty **then break**
  $Q_n = \{\}$
  for $P \in Q$ **do**
    flag = false
    for P' in parents(P) **do**
      if $P' \notin Q_p$ or $P' \in \mathcal{M}$ **then** flag = true; **break**
    end for
    if flag **then continue**
    $cnt = cov(P, \mathcal{D})$
    if $cnt < \tau$ **then**
      add P to $\mathcal{M}$
    else
      add children of P based on Rule 1 to Q_n
    end if
  end for
  $Q_p = Q$; $Q = Q_n$
end for
return $\mathcal{M}$

Figure 6. Pseudocode of Pattern-Breaker (top-down algorithm). Graph from Abolfazl [2].

4.2. Nutrition Label Generation

4.2.1 Functional Dependencies

Functional Dependencies [7] are collections of attributes that uniquely determines another attribute. This is a big drawback is our modern data set available publicly. When an ML algo classifies something which is not remotely close to the outcome predicted. This can have a deep impact in real-world applications. FD is a way to know if any set of attributes is directly related to another attribute of our dataset. That way data-scientists know which attribute to ignore/modify. This is a star relational tree diagram based on adult census data table [6]. This figure show the attribute(SEX) in this particular data-set is uniquely influenced by all the other subset combined(NODE) available(e.g. (income,occupation,age,hours per week) combined determines the attribute SEX) and many others. Every node in this figure determines a relation between the attribute SEX and the node title. So the SEX attribute is probably biased towards specific age with specific income

Figure 7. Star relational tree diagram.

with people who works specific hours per week, we must balance the dataset by modifying or use some sort of feature engineering.

4.2.2 Association Rules

Association Rules as the name implies , when a set of attributes directly determines another set of attributes available in our dataset. This is another tool for our data-scientist to determine whether he/she should modify or ignore attributes. Make decisions that will help the ML algo to predict the right outcome. Meta-Data generation: For meta-

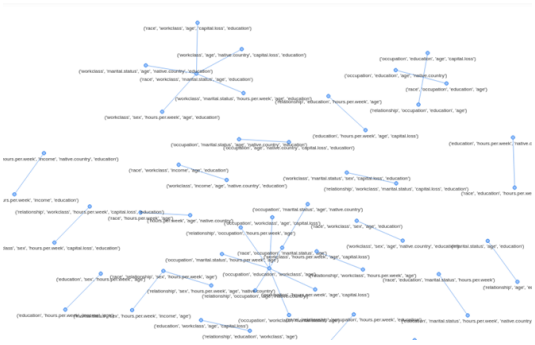

Figure 8. Multiple star relational tree diagram.

data generation we use python library called pandas profiling. We pass the data-set to the class and use ProfileReport function to generate basic meta-data of the data set (e.g. attribute distribution, correlation between multiple attributes, missing-values that may be available in the data-set etc.)

Exacting Biases: In order to extract biases available(if any) , we tried the method of finding functional dependencies of the data-set, then plot the subsets.

We use the similar method the way we find FD(functional dependencies). We read the data-set using Pandas library. Take counter variable k=0 and a list U , which contains attribute names of the data-set. Then we find a singleton lattice variable C for every item in the list U and append that with None value for each item in the range of length of U – 1.

$$C = \sum_{i}^{i=U}[i] + \sum_{j=0}^{j=len(U)}[NULL]$$

Then we generate k-level subset attribute using our powerset(PS) function. Then use the subset to make a python dictionary of binary representation of every k-level subset available.

$$PS = \sum_{i=0}^{1*len_2(U)} \sum_{j=0}^{j=len(U)} U[j]$$

Then we initialize the closure dictionary with the next k-level and update the dictionary, and we also update the cardinality dictionary for the next k-level. If k is not 1 we calculate dereference Closure and Cardinality,we go on to reduce the k-level candidate row from k-1 level.

Then we use our GetFDs and Obtain Equivalences function to get subsets from the Closure and Cardinality.

5. Result

	Dataset_ID	Age	Gender	Race	Ethnicity	Country	Maximal Uncovered Pattern	Uncovered Attributes	Report
0	2a55-dhk8	Not Exist	Not Exist	Not Exist	Not Exist	Not Exist	[]	nan	2a55-dhk8.html
15	2cie-ugkk	Exist	Not Exist	Exist	Not Exist	Not Exist	[['Skykomish School District No. 404 Skykomish School District No. 404 Proposition No. 1', ['Snoqualmie Valley School District No. 410 Snoqualmie Valley School District No. 410 Proposition No. 2', '16-24'], ['Northshore School District No. 417 Northshore School District No. 417 Proposition No. 2', '35-44'], ['Lake Washington School District No. 414 Lake Washington School District No. 414 Proposition No. 3', '25-34'], ['Shoreline School District No. 412 Shoreline School District No. 412 Proposition No. 2', '45-54'], ['Shoreline School District No. 412 Shoreline School District No. 412 Proposition No. 1', '55-64'], ['Fife School District No. 417 Fife School District No. 417 Proposition No. 1', '>64'], ['City of Black Diamond Ballot Synopsis of Recall Charges Against Councilmember Patricia Pepper, City of Black Diamond', '	('Race', 'SumOfCount')	2cie-ugkk.html
16	2cjd-uvx7	Not Exist	Not Exist	Not Exist	Not Exist	Not Exist	[]	nan	2cjd-uvx7.html
17	2cma-9y6y	Exist	Not Exist	Not Exist	Not Exist	Not Exist	[['35-44'], ['25-34'], ['45-54'], ['16-24'], ['55-64'], ['>64'], ['	('AFF_UNITS_NET', 'UNITS', 'DELETE - Police Districts', 'SUPE_DISTRICT', 'SF Find Neighborhoods', 'DELETE - Neighborhoods')	2cma-9y6y.html

Figure 9. Generated result from Metadata Generation.

The above is the generated metadata. First attribute is the dataset ID, 'Age', 'Gender', 'Race', 'Ethnicity', and 'Country' are sensitive attributes. Values for these 5 attributes are 'Exist', 'No exist', a special value for 'Gender' is 'Predict'. 'Predict' is that if there is no gender attribute, but this is name attribute, we will predict gender based on name. Then is 'Maximal Uncovered Pattern' and 'Uncovered Attributes' indicates MUP and which attributes are uncovered. 'Report' is a link that will open a complete report on the data table. The report contains number of attributes, the type of attributes, percentage of missing value and total size in memory and percentage of duplicate rows. For each variable, it will count number of distinct value, the percentage of distinct values, and the distribution plot. Also, the value distribution among attributes and correlation among attributes. The report is very complete on single attribute and between two attributes.

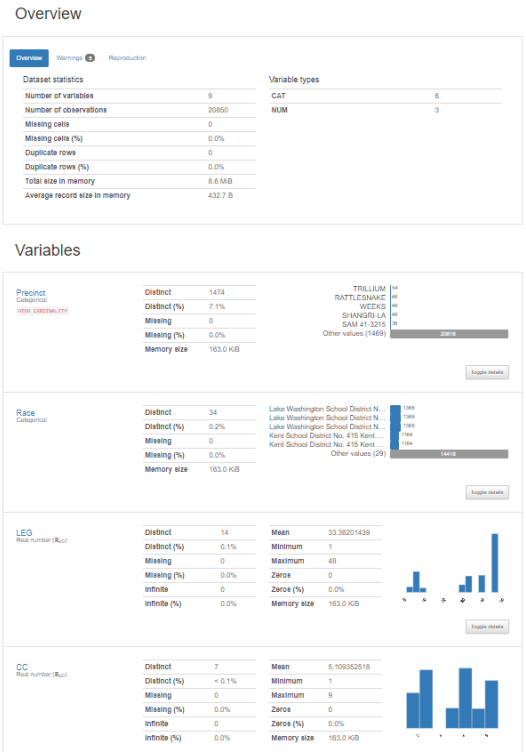

Figure 10. Example of report.

6. Future Work

Current problems are: (1) Long time computation and memory usage. It takes about 2 hours to generate reports for 183 data tables, which there should be further optimization on algorithms, we could use deep-diver algorithm [2], or sampling algorithm to save time, and sometimes we find 32 GB memory is not enough, it will interrupt the process. (2) Better visualization: We have tried to plot the pattern graph for each data table. The result graph is clear if there are only countable value combinations. During the test on adult census data table [6], the pattern graph contains about 800 nodes, even we plot it, it is not clear as we expect. (3) A website might be a good format as output. Also, a search function might be helpful for users. A search function like what Aidan is doing can help makes this system complete. Users not only can find datasets, but also they can choose datasets based on information we provide.

7. Reference

[1] M. Mulshine. A major flaw in google's algorithm allegedly tagged two black people's faces with the word 'gorillas'. Business Insider, 2015. networks." Advances in neural information processing systems. 2014.

[2] Abolfazl Asudeh, Zhongjun Lin, H. V. Jagadish. Assessing and Remedying Coverage for a Given Dataset, 2019.

[3] Seymour Sudman. Applied sampling. Academic Press New York, 1976.

[4] Madelon Hulsebos, Kevin Hu, Michiel Bakker, Emanuel Zgraggen, Arvind Satyanarayan, Tim Kraska, Çağatay Demiralp, and César Hidalgo. 2019. Sherlock: A Deep Learning Approach to Semantic Data Type Detection. In The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330993

[5] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: Risk assessments in criminal sentencing. ProPublica, 5/23/2016.

[6] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[7] Chenkai Sun, Abolfazl Asudeh, H. V. Jagadish, Bill Howe, Julia Stoyanovich. MithraLabel: Flexible Dataset Nutritional Labels for Responsible Data Science, 2019.

8. Github

Link: https://github.com/Tianyu9748/Data-Profile