Abstraction Layer for Data Cleaning Tools

Milad Abbaszadeh Jahromi[1]

[1]Technische Universität Berlin

author1@tu-berlin.de

Abstract. *This is the abstract... Mohammad: leave abstract. i will write it.*

1. Introduction

1.1. Motivation

This report talks about the abstract layer which can help users to avoid from conflicting with the tools for cleaning purpose and help them for unique and constant input and output which can be really remarkable to save time for people who wish to work with these tools. Mohammad: the motivation can be so much better and stronger. talk about different tools that have heterogeneity in terms of language, installation, input/output system and so on and the user difficulties.

1.2. Contributions

If you have a file and wish to clean it with cleaning tools and you wouldnt like to get involved in installation and command usage which is required for running, you can simply use our abstract layer. With this layer you can simply send your file and then get the Suspicious element by the constant type and form in output for each tool. Mohammad: talk about how this module can address the so-called heterogeneities for the user's application. use bullet list just like the other papers' contributions. talk about user instead of "you".

1.3. Architecture

Mohammad: in this subsection talk about the big picture of the architecture and the position of the abstraction layer in the user system. I have added a proper figure. you write the texts.

1.4. Overview

The rest of this document is organized as follows. In Section 2 we explain how the user can install the module. In Section... Mohammad: complete this subsection

2. Installation

2.1. Prerequisites

To install the module, you need first to install followings:

- Linux (Ubuntu/Debian recommended)
- Python 2.7
- Oracle Java 1.8
- Apache Ant 1.8.2+
- Postgres SQL 9.2+

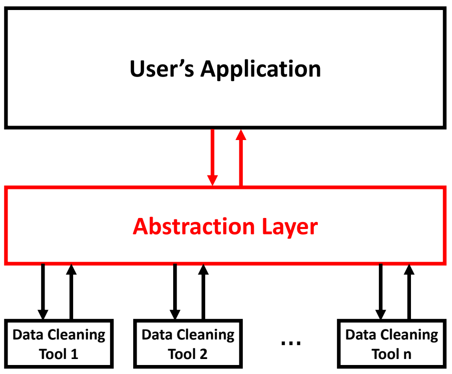

Figure 1. The position of the abstraction layer in the architecture of the user's system.

2.2. Setup

Mohammad: talk about how the user can use the installation function in the first place to installs and configures the tools. introduce the API function that installs and configures the tools.

3. Start to Work

3.1. Input

For starting with abstract layer you should make sure that you make dictionary like dictionary that is introduced below as input for our layer. This dictionary has two keys that you can set the first one with the type of your file and the location of the file as path. In fact, for now the layer only support the files that generate with csv format but we will keep going to add other formats for future. Mohammad: before introducing the template of your input configuration (look below), first start to talk generally about: (1) proper api function. (2) define the argument that it needs (input configuration, data cleaning job, or whatever you call it), then you have the following codes:

```
run_input = {
        "dataset": {
                "type": "csv",
                "param": ["the/address/of/your/file.csv"]
        },
        "tool": {
                "name": "tool_name",
                "param": ["list_of_parameters_wrt._to_the_tool_name"]
        }
}
```

Mohammad: after the general template of the input object, start to explain each key and its proper values in a nested list:

- **dataset**. This part of the input configuration is responsible to describe the input dataset. it consists of two options:

  - **type**. This option specifies the type of dataset. Currently the module only supports ”csv” datasets.
  - **param**. This option specifies the list of parameters that are needed for accessing the dataset. For csv datasets, this list contains only the path of dataset.

  Mohammad: complete this structure...

Mohammad: after that you can provide some examples.

3.2. Output

You will get one list as output that you can find it below [row, column, suspicions value]
Mohammad: just like the input subsection, write in this subsection.

References