
Internship Report

Luwei Wang in Data 4 Common Good

Name: Luwei Wang

Student ID Number: N17988219

Degree Program: MSIS

Internship Semester: 2024 Summer

Employer: Data for the Common Good

Employer Location: Chicago, IL

Supervisor: Luca Graglia

Supervisor Email: lgraglia@bsd.uchicago.edu

This report is submitted to NYU's Computer Science Department for the sole purpose of assigning an internship grade. The information remains confidential and proprietary to the company.



Abstract:

The report provides an overview of Luwei Wang's internship in Data for the Common Good organization through the Google Summer of Code program during Summer 2024 semester. The article introduces the D4CG organization and clarifies the development process of the Self-generated Table 1 project by discussing the user's requirements and problems that occurred during development. Provided technical solutions at each stage and described the gains from the internship experience.

1. Introduction

Employer Introduction

Data for the Common Good (D4CG) is headquartered in the Department of Pediatrics at the University of Chicago. The organization is dedicated to building communities, platforms, and ecosystems that maximize the potential of data to drive discovery and improve human health. The team comprises medical experts, data analysts, and computer science engineers. The team also works with collaborators all over the world to connect and share useful, high-quality data between institutions, groups, and countries to increase opportunities for discovery.

The organization currently runs several projects that are making a difference in the areas of medical data collection and clinical decision support. One of the projects is the "GEARBOX" tool. It was developed by D4CG in partnership with The Leukemia & Lymphoma Society, which is currently in use for patients with acute myeloid leukemia and will be expanded for use with other cancer types.

The signature project of the organization is the Pediatric Cancer Data Commons (PCDC). PCDC is a data collection and analysis project, gathering pediatric, AYA, and adult cancer clinical data from around the world. Along with data collection, the PCDC Data Portal provides a window for the researchers to access the data straightforwardly. The Data Portal also integrates analysis tools for the users to explore the data. [1]

Internship Overview

By being accepted by Data for the Common Good through Google Summer of Code, I completed a three-month internship in the Summer 2024 semester from May to August. The work I've done during the internship is about upgrading the PCDC portal by developing a new component for the data analysis tool. The former PCDC Data Portal has provided APIs for data access (downloading datasets in the JSON format) and filtering by tabs. Based on these APIs, the upgraded version should provide a new Self-generated Table 1 tool to further support the researchers' data exploration process.

The demographics table, so-called Table 1 is an important feature in medical-related scientific research. Table 1 typically presents an overview of various demographic and baseline variables of the study population, and always appears as the first table in the paper[2]. While Table 1 is necessary for the articles, the generation process is complex. The traditional method requires statistical software knowledge and is quite time-consuming. What's more, it's also tedious to meet the formatting standards. However, since the PCDC is an open and extensive dataset library, if the portal can provide self-generated Table 1, it will significantly improve the efficiency of the researcher's work.

During the Internship, I worked with the PCDC engineer team. Under the supervision of my mentor Luca Graglia, who is the Director of Software and Infrastructure Services of Data for the Common Good organization, following the goal of developing a usable and stable application for users to generate Table 1 from the existing datasets in the Pediatric Cancer Data Commons library. I worked on both the front end and the back end and accumulated a wealth of technical experience, deepening my knowledge of web application development.

2. Discussion of Projects

2.1 Requirements Description

Demographic tables are used to set the baseline characteristics in medical research. The tables provide insights into variations in demographic attributes within the population based on treatment, exposure, or outcome. Such a table usually includes summary statistics and P values. Summary statistics often contain counts, means, standard deviations, medians, 25th and 75th percentiles, and ranges for continuous variables.

As for the categorical variables, the tables convey the statistics like frequencies and distributions. Categorical variables may combined with the continuous variables, calculating the bucketized statistic results.

The goal of the project is to develop a usable and stable application for users to generate Table 1 from the existing datasets in the PCDC library. The application should be consistent with the style of the whole platform and follow the original coding structure. The application should achieve the following functions:

- Grouping Variable Selection: The grouping variables should align with the PCDC Data Portal analysis tool user filter archives. Adapt appropriate components to help the user's selection.
- Variables and attribution selection: After identifying the grouping filter, the user will select the attributes to be included in the table's main body and the summary statistics method for each attribute. The users can also rename the items and label the attributes by groups. The selection has constraints based on the configuration features of the option tabs. Current methods include logical and, logical or, and value comparison.
- Table displaying and saving: The user can get an overview of the statistics and have a way to save the result.

By implementing the above functions, the users can set all of the features for an effective Table 1. Based on the user's options, the completed platform should successfully generate qualified Table 1 in proper formats.

2.2 Technical Details

Project Overview

To achieve the goal listed in the former section, I worked on both the front end and the back end. The technology stack for the front end is React.js with Redux status management tool. For the back end, the project adapts Python with Flask. The communication between the front end and the back end is through HTTP protocols with REST APIs and GraphQL. REST APIs will handle the analysis request like generating Table 1 and requesting survival plots. GraphQL is used for direct query of specific data.

GraphQL offers advantages over REST APIs involving data fetching. It's easier to maintain since it works on a single endpoint and enables the user to request all the data they need in one request. These advantages make it a good choice to handle the data retrieval.

The portal involved access authentication and authorization with JWT and signature. During the whole internship process, I worked on an AWS client and Github to control the version.

Prototype Stage

Before I began working with the code of the currently deployed project, I first worked on a simplified prototype of the self-generated Table 1 function. The prototype used fake data with limited basic features. It had the function of grouping variables and covariates selection and rename. The prototype consisted of two parts: front-end form and back-end micro-service, mimicking a simple workflow of Table 1 generation with categorical, continuous, and bucketized variables.

At the prototype stage, I mainly solved two problems: cleaning the query data and the initial design of the request APIs.

Before doing statistics and generating Table 1 data, first need to download the required data from the database with the Guppy service. The downloaded data is in

the format of embedded JSON. For the data cleaning, I solved the problem with the following steps:

1. Handle missing values and align the columns. The data from PCDC has different data sources. The sources format the data differently and contain different aspects of a subject, so the cleaning process is necessary. First, handle the missing data. Replace the NaN value with "Not Reported". Then align the columns following the configuration files.
2. Unfold the embedded structure. Since the data format is embedded JSON, some attributes have an array or an object as a value, such as Secondary Malignant Neoplasm. One subject may get multiple times SMN throughout their lifetime. JSON format data needs unfolding to work with Pandas library while keeping the index of the subject.
3. Organize the columns and assign proper names.

When the data was ready, I made a draft of the APIs working locally on the prototype. From this step, the main purpose is to clarify what values are needed for both ends. The final result could generate Table 1 supporting multiple grouping variables and covariables selection and rename.

The prototype project uses React.js and origin HTML components for the frontend, Flask, Pandas, and NumPy libraries for the backend. Contributed approximately 600 lines of code in total.

Production Environment Stage

After the function worked out on the prototype, I hand on the real project in the production environment. The project consists of two repositories: Frontend Data-Portal (<https://github.com/chicagopcdc/data-portal>) and Backend Analysis Tool (<https://github.com/chicagopcdc/PcdcAnalysisTools>).

First, I set up the environment for development. The development server is in UChicago's intranet, so most of the work was completed on an AWS server, and I have the SSH token to access the instant. However, the authentication involves the hostname. To set the debugging environment for the front-end project, I set up a double proxy with Nginx and successfully mimicked the intranet environment.

After that, I start transferring and adapting the prototype functions to the production projects. Unlike in the prototype, the data in the real project has many more attributes and the variable selection workflow needed modification. In the prototype, we allow the users to explicitly determine the grouping variables, while in the real project, to align with other parts in the analysis tool, the users should set up filters as a grouping variable. The filters are saved and loaded per client, so more API calls are needed and I updated the request formats. Since the filters are customized by the users, which means the filters may not be strict partitions on the whole dataset, the component tightens the restriction that there can be only one filter as the grouping variable.

The frontend project uses the Redux state management library. With Redux, the project behavior is much more consistent and predictable and the data management is easier as well. PCDC data portal uses its own component library, I also learned to write component libraries. The main difficulty in the frontend development is to understand the Redux behavior so that I can fetch the required data correctly. The functionality of the program is well encapsulated and highly modularization. What's more, I should keep my component behaving consistently with the former parts of the portal. The way I solve the problem is to track the data flow and draw layer maps.

The backend project uses docker to wrap the development environment. There are two main difficulties during development. One is the authorization process. This is also caused by the intranet environment. I made fake JWTs and bypassed the checkpoint. Another one is caused by the extended attributes. When working on prototypes the statistics only involved basic features like name, sex, and race. In the real project, the attributes are much more complex, which means a more complex data cleaning and preparation process. I went through the attributes and classified them by their features. For each class, I carried out different cleaning rules.

The project is still being developed. The following parts should be further modified:

- Enhance the table display. Provide more layout templates for users to choose
- The current project only covers a part of the attributes. Should include more features in the filter table.
- Keep longtime maintenance when PCDC continues to grow.

3. Summary and Insights

After the three-month internship, I benefit a lot. First of all, D4CG is an organization with open and friendly circumstances. I get closer to the workflow by working with my mentor Luca and colleagues. It also provided me an invaluable opportunity to learn from and hand on complex real-world web applications. I felt the progress in my development skills.

The former academic course “Data Science for Business” helped me a lot, since there are many works related to data preparation. The concept of cleaning and general methods of data cleaning helped me get a better understanding of the data and handle the statistics more smoothly.

One lesson I learned from the experience is that there are no ‘fields’ in the industrial production environment. During the internship, some of my work may be in the field of UI/UX, and some of my work may be in the field of data science. However, the main purpose is always to find a solution. Even when it comes to the problems related to software engineering, it still needs the ability to collect information and continuous learning to optimize the whole user experience. I believe that this internship experience will keep helping me when I face more problems in the future.

References

[1] <https://commons.cri.uchicago.edu/>

[2] <https://exploratory.io/note/kanaugust/How-to-Create-Table-1-Summary-Information-YYf5BPf4Xs>