Emily Cunningham
INFO 628: Data Librarianship and Management
Fall 2024
Final Project Check-in #2

Analysis Writeup

This semester I started off thinking I would use the BOP Federal Inmate Complaint Dataset to do more more advanced statistical analysis using R, but during the long process of making the dataset more usable for myself, I switched gears to focus more on making the dataset more usable for the general public.

I am continuing to focus on NY submissions, but I am adding inline comments to the Python script that could enable someone with basic coding knowledge to create a similar enriched and expanded dataset for another state.

Enriching the Dataset
I have enriched the original dataset by adding several columns that make it easier to understand the complaint more holistically, taking each submission into account. The subcount column contains the number of total submissions associated with that one case number. The rejcount column counts the total number of rejected submissions associated with that one case number. The cldclocount column counts the total number of submissions with a “Closed Denied” or “Closed Other” status update for that one case number. The clgacccount column counts the total number of submissions with a “Accepted” or “Closed Granted” status update for that one case number. These few columns can help users better understand the history of the complaint, without having to sort by case number to count the number of submissions. I also added three columns that help users better understand the time frame of the complaint from beginning to the most recent status update. The earliestsitdtrcv column shows the earliest submission date for all submissions with the same case number. The latestsdtstat column shows the latest date a status was assigned for all submissions with the same case number. The daysbetween column calculates number of days between earliestsitdtrcv and latestsdtstat, which shows users how long an inmate spent trying to get their complaint resolved.

Expanding the Dataset
As I was initially evaluating the data I had a really difficult time understanding what I was looking at because of the various codes used as column headers and values. I decided to create an expanded version of the dataset with translations of all of the codes for my own personal use, but then I thought that if an expanded dataset is helpful for me, it will likely be helpful for someone else as well. I included cleaned up code dictionaries and the python script which creates the expanded dataset.

Creating a Subset of Unique Complaints
When I was starting to do some basic summary statistics on the dataset, I realized that if I was trying to determine which complaint type was the most common in NY facilities between January

2000 and May 2024, my data was going to be skewed. Many complaints have more than one submission associated with them, and some complaints have over 10 submissions. I needed to create a subset of the NY submission data that contains only unique complaints. I chose to only include the submission with the most recent status update date for each case number. With the addition of the subcount, rejcount, cldclocount, and clgacccount columns, I could get a pretty good idea about the history of the complaint, just by looking at the most recent submission.

I also decided to make an expanded version of the unique NY complaint dataset with translations of all of the codes for the columns and values.

Summary Statistics and Visualizations

When I finally got the datasets to a good place where I could understand the data and start to think about what questions I could ask of the data, I decided to change directions with the statistical analysis. I first wanted to do some more advanced statistical analysis, but I thought that if my data cleanup was more geared towards helping the general public view the data, then I would like to gear my analysis towards that same audience as well. I decided it would be more in line with my goal to instead try to create some visualizations based on simple statistical analyses to make the data easier to understand and to showcase what kinds of questions users could ask in regards to other states/regions. I have started playing with R to make some of these visualizations, but the learning curve feels quite steep. I may switch my approach and use either the Matplotlib or Plotly Python libraries.

Questions I will create visualizations for:

- What is the average number of rejected submissions per complaint case number?
- What are the most common “status reasons” for rejected submissions?
- What percentage of unique complaints only have rejected submissions, i.e. the complaint was never properly addressed and therefore the inmate never received any resolution or explanation for why the issue could not be resolved.
- Which complaint types had the most “closed denied” or “closed other” status updates, i.e. which complaints were most likely to be denied.
- Which complaint types had the most “closed granted” or “closed accepted” status updates, i.e. which complaints were most likely to be granted/resolved.
- Which complaint types have the largest average daysbetween value?
    - For each of the top 10 complaint types, what percentage are “closed denied”/”closed other” vs “closded granted”/”accepted” vs only “rejected” without being closed?
- Which NY facility has the largest average daysbetween value?

Data Management Plan (v2)

Storage and Backup Procedures

Following the 3-2-1 rule, data will be securely stored in three separate locations including Google Drive, the desktop of my laptop, and a GitHub repository. This approach will ensure data redundancy.

File Formatting and Naming

All raw data will be stored as CSV files. Python scripts and R scripts will be saved as Jupyter Notebooks (.ipynb) in order to offer researchers inline documentation, which should enhance the clarity and reproducibility of the code. Scripts will also be saved as .txt files, to ensure long term accessibility, in the case that .ipynb files become unopenable in the future. Data visualizations will be saved as PDF files.

Files will use PascalCase and will be named logically. Each file will have a number prefix (01_, 02_, 03_, etc), which will help me and future users to understand the order in which the various documents are used/created. If files need versioning, they will keep the same number prefix and title, but a version number will be appended to the end of the file name (e.g., _v02). Files names will be kept short and will not have spaces or special characters. All files are currently nested in one single folder. This may change as the project progresses.

Files will also be versioned in their GitHub repository, where changes in code can be tracked and modified. I will write more descriptive commit messages that can help others and myself.

Project Organization and Structure

I hope to keep my number of files fairly low, so that they can all live in one project folder. My use of number prefixes in the file names should help future users better understand the workflow organization and streamline its future use. As the project's complexity potentially increases, I may switch to an organizational scheme that includes four subfolders (src, data, results, docs).

Documentation

Scripts will be saved as Jupyter Notebooks so that markdown cells and inline comments can serve as a form of in-depth documentation.

A README file will be created for the GitHub repository, which will outline the entire project, and provide context as to what each file is, where it came from, and how it can be used. Jupyter Notebooks will be uploaded with their output, so that viewers can see the intended output and compare their results.

The Data Liberation Project provides some codebooks for the column headers, facility codes, and complaint subject codes. Some of these will have to be cleaned up a little bit. The raw codebooks along with my wrangled versions will all be included in the project folder.

I will also keep a research journal where I will keep track of each step taken during the research process, that way I can assure that the README file has information about various programs and settings that are necessary in order to get the scripts to work properly.