NCI DCEG Internship Challenge

Here's a data analysis challenge for you that is similar to the stuff we do at DCEG. Like most problems in data science, there is no single solution, and the effectiveness of yours will largely depend on your understanding of the data. There is no time limit, but it shouldn't take you more than 4-5 hrs to design & implement your solution.

Problem Statement

We need to analyze the various Disease Indicators behind Cancer and display our analyses in the form of visualizations on a web dashboard. To do this, we shall leverage the deidentified Hospital Inpatient Discharges dataset made available by the state of New York. The dataset can be found here:

https://health.data.ny.gov/Health/Hospital-Inpatient-Discharges-SPARCS-De-Identified/gnzp-ekau

and the raw data can be found at the below URLs:

JSON: https://health.data.ny.gov/resource/gnzp-ekau.json
CSV: https://health.data.ny.gov/resource/gnzp-ekau.csv

These URLs only return 1000 observations at a time by default, but the entire dataset contains over 2 million observations of patients, including their characteristics, diagnoses and treatments. It has data across medical ailments, but since we wish to perform our analyses only on cancer diagnoses, we can reduce the size of the dataset by applying a filter like below:

https://health.data.ny.gov/resource/gnzp-ekau.json?$where=UPPER(ccs_diagnosis_description) like '%25CANCER%25'&$limit=10
(Check the Examples section to understand how this URL is formed.)

This is actually a REST API that allows you to efficiently query the dataset. There are several such operations that you can run on the data merely by changing the query parameters in the API. More Documentation here:

https://dev.socrata.com/foundry/health.data.ny.gov/gnzp-ekau

The first thing you should do is to `Sign up for an app token` on the page that opens up. This token will allow you to make calls to the API without having to worry about being restricted. The documentation contains more info about how to use the token, along with helpful information about the columns in the dataset.

All interactions with the API should be made within the code itself via HTTP requests. In other words, DO NOT download the data as a file locally, but call the API in your code, and use the response directly as your dataset.

Your task is to use this dataset and create meaningful visualizations that show inherent relationships within the data, be it in the form of histograms, line graphs, pie charts or any other you deem suitable for the analysis. These visualizations should be part of a webpage that can be seen in a browser.

Submission

Your code must be available as a GitHub repository, and you need only send us the URL to the repository with your application. For serious bonus points, you can host the dashboard you create on a free hosting service (such as GitHub Pages or Heroku) and add the URL for where it is hosted to your README.md file on GitHub.

Evaluation Criteria

- **Please note again that this is meant to be a short exercise**, so do not spend too much time on cleaning the data.
- Although using Python/Pandas or other languages and frameworks is not a dealbreaker, we highly recommend using JavaScript to implement your solution. Libraries like plotly.js, Chart.js should make it easy for you to design your visualizations directly for the browser.
- Your code must include: a) interactions with the API and the queries you used, b) any filtering of the data you did yourself, and c) how you generated the visualizations from that data.
- The quality of your code and the analyses you run will be what you will be primarily evaluated on. Interactive visualizations will earn extra credit!
- Do not fret too much about making it pretty -- color schemes, or any CSS other than the most basic stuff will not carry any points. We will only test it on Chrome, so don't worry about browser compatibility either.
- You should not need to write any backend code for this problem, but if you feel you need a server running, you should include the code for it with your submission. Be sure to mention this in your README, along with the instructions on how to run it.

Examples

- **Using the API**: You might have noticed that the base URL for the raw dataset and the one we used to reduce it to include only cancer patients is the same -- all that was added to the URL in the latter case were the query parameters. There are 2 query parameters that we used in this case:

  ```
  $where - defines the field(s) and the condition(s) to filter the data on
  $limit - defines how many observations to return
  ```

  We used the **$where** clause to check for the presence of the word "cancer" in the patient's diagnosis (the *ccs_diagnosis_description* field in the dataset), and the **$limit** clause to get back only 10 results. Try changing the value of $limit to 50, or replace the word "cancer" with "pneumonia" in the URL.

  Note: You could of course write your own code to perform querying on the entire data yourselves, which would be completely fine by us. However, this is meant to be a fairly low-effort challenge, so we recommend that you use the functionality provided by the API as much as possible, and only write your own querying functions when the API cannot provide you what you need.

- **Visualizations**: Examples of some basic analyses you could run might be:

  - Mortality rates across age groups and cancer types
  - Frequency of cancer diagnoses by gender and/or ethnic backgrounds
  - Correlations of specific types of cancer with age groups

  None of these are mandatory, and you may choose to create your own. There is no limit on the number of visualizations. Anywhere between 2 and 5 should be good -- just ensure that they are not repetitive.

For any other questions related to the challenge, please email us at bhawsarpm@nih.gov.