NCI DCEG Internship Challenge

Here's a data analysis challenge for you that is similar to the stuff we do at DCEG. Like most analytical problems in data science, there is no single correct answer, and the effectiveness of your solution will largely depend on your understanding of the data. Note that there is no time limit to solve the problem, but it should not take you more than 4-5 hours to design and implement your solution.

Problem Statement

We need to analyze the Disease Indicators behind Cancer across age groups, genders, and racial and ethnic backgrounds 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

This dataset contains over 2 million observations of patients, including their characteristics, diagnoses and treatments. It has data across medical ailments, but since we only wish to perform our analyses on those who were diagnosed with cancer, we can considerably reduce the size of the dataset by using this as our base data instead:

https://health.data.ny.gov/resource/gnzp-ekau.json?$where=UPPER(ccs_diagnosis_description) like '%25CANCER%25'&$limit=10

You might have noticed that the base URL for the raw dataset and the one for the reduced dataset is the same -- all that was added to the URL in the latter case were the query parameters. This is actually a REST API that allows you to efficiently query the dataset. There are 2 query parameters that we used in the above case:

```
$where - defines the field(s) and the condition(s) to filter the data on
```

```
$limit - defines how many observations to return
```

Since we wished to get the data of only patients diagnosed with cancer, 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). There are several such operations that you can run on the data merely by changing the query parameters in the API. You can find more documentation on what these are and how to use them 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 (more info about how to use the token on the page). This page also contains helpful information about the columns in the dataset: for instance, the Age Group column must be referenced as 'age_group' when using it in the API query parameters, and is of the type ***text***. You will also find some example API calls that show you how to perform filtering and querying on the data directly using the API.

You might find that there are a few fields in even the reduced dataset that are irrelevant to your analyses (such as perhaps the *operating_certificate_number*), and you may choose to disregard those fields or remove them from the base dataset you use in your analysis.

Note: You could of course write your own functions 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 leverage the functionality provided by the API as much as possible, and only perform your own queries when the API cannot provide you what you need.

Examples of some basic analyses you might want to run could be:

- Mortality rates across different types of cancers
- Prevalence of cancer diagnoses among people with various racial and/or ethnic backgrounds
- Frequency of the different types of cancers within age groups, and correlation of specific types with specific age groups

None of these are mandatory, and you may ***(and should)*** very well choose to create your own based on the data.

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 bonus points, you can host the dashboard you create on a free hosting service (such as Heroku) and add the URL to your README file on Github.

Evaluation Criteria

- **Please note again, that this is only meant to be a short exercise**, so please do not spend much time cleaning the dataset or making the dashboard beautiful -- we shall not evaluate your submissions based on the prettiness of the dashboard.
- 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 on the browser.
- Code quality and the analyses that you run will be what you will be primarily evaluated on. Interactive visualizations will earn big ups!
- You should not need to write any backend code for this problem, though if you feel you need a server running, you are free to 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.