Healthcare Analytics – Recommender system for hospitals based on Medicare ratings and patient surveys

Scenario

The US government has two main single-payer, national social insurance programs. Medicare is for seniors who are age 65 and older who paid into the Medicare system when working (or their spouses paid into the system), and also for persons eligible for disability. Medicaid is for those who are indigent, and for seniors who never worked and did not pay into the Medicare system (or did not have a spouse who paid into the system).

In the US, most hospitals are private, and private hospitals can be for profit or not for profit. Some hospitals are public, owned by a local county government, and often referred to as a "county hospital." Few counties have more than 1 hospital. Most large cities have several private hospitals, a mix of for profit and non- profit.

For most hospitals, Medicare / Medicaid is a major source of revenue, and for county hospitals it is usually the main source of revenue. Medicare/ Medicaid has standards that hospitals must meet. Some payments are withheld if standards are not met. Some payments are bonuses if standards are exceeded.

The main data sets are available for download at the official government website: data.medicare.gov. The data set of interest to us is the Hospital Compare data. This data set is updated several times per year. Private hospital owners and local county governments are always very much interested in this data set, as a major source of their revenue depends on meeting the criteria. Since hundreds of billions of dollars of government payouts are based on this dataset, it is widely studied, and there are numerous consultants and consulting businesses built around this dataset. Most data science shops associated with hospitals routinely analyze this data set.

Big Data is often described in terms of the 3 V's: Volume, Variety, and Velocity. In this assignment, we will focus on Variety. (One remaining assignment will focus on Volume, and the other will focus on Velocity). We will both read and write data in a variety of ways. These are common, everyday work for a Data Scientist.

We have a proprietary in-house system that creates our own ranking of hospitals and a list of focus states for analytics. The file produced will be an MS Excel workbook with 2 sheets. One sheet will have a ranking of hospitals and the other sheet will have a list of focus states. Your Python code will need to download this spreadsheet and read the data for use in further analytics.

You will perform analytics using the data you loaded into SQL and produce 2 MS Excel Workbooks.

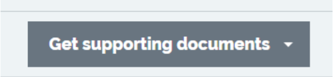
The first workbook will have hospital ranking information. It will have 1 sheet with the top 100 hospitals nationwide. For each of the states in the focus group, it will have the top 100 hospitals for that state.

The second workbook will have a statistical analysis of the measures used to determine hospital ranking. It will have 1 sheet with each of the measures, along with the minimum, maximum, mean, and standard deviation for that measure for all hospitals. For each of the states in the focus group, it will have the same statistics for each measure, but only for hospitals in that state.

Downloading the Medicare Hospital Compare Data and Loading it into SQL

Visit the website, data.medicare.gov, click on the “Hospital Compare data” graphic.

In the upper right corner, you may want to look at the “Get supporting documents” dropdown, especially the “Downloadable Database Dictionary” which explains more about this data set.



Download the zip file of the Hospital Compare datafiles using the following link
(it is all one line – word cannot fit it into 1 line):
https://data.medicare.gov/views/bg9k-emty/files/0a9879e0-3312-4719-a1db-39fd114890f1
?content_type=application%2Fzip%3B%20charset%3Dbinary&filename=Hospital_Revised_Flatfiles.zip

Your Python program will need to create a staging subdirectory called “staging”. Use only relative path names and use the Python machine independent joining of directory names.

Your Python program will need to unzip the file into various csv files in the staging directory. The first line of each file will be the list of fields. Using the staging directory will make it a lot easier to look at the files for debugging purposes.

Your Python program will need to create an SQLite database in the local directory (not in the staging directory – don’t use any path names) named “**medicare_hospital_compare.db**”.

Your Python program will need to create a table for each csv file in the staging directory. You should ignore files other than csv files. Note: in this data set, there is one file that is corrupt, which you may ignore: "**FY2015_Percent_Change_in_Medicare_Payments.csv**"

Table names should be the same as the file name without the file extension (.csv), with the transformation detailed below applied.

Column names should be the same as the field name in the first line of the csv file, with the transformation detailed below applied.

Table names and column names should have the following transformations to make them acceptable for SQLite (in this order):

1. Convert all letters to lower case
2. Replace each blank “ “ with an underscore “_”
3. Replace each dash or hyphen “-“ with an underscore “_”
4. Replace each percent sign “%” with the string “pct”

5. Replace each forward slash “/” with an underscore “_”
6. Multiple underscores in a row are ok – actually needed in some cases to prevent duplicate names
7. If a table name starts with anything other than a letter “a” through “z” then prepend “t_” to the front of the table name
8. If a column name starts with anything other than a letter “a” through “z” then prepend “c_” to the front of the column name

Since we are loading these as staging tables, use “text” for the data type for every column, even if you think the column might be numeric, a date, etc. This will allow anything to be loaded into every column, otherwise bad data would cause load errors.

All data from each file should be loaded into the corresponding table, except the first line containing the field list.

Download MS Excel Workbook of In House Proprietary Hospital Rankings and Focus List of States

Our in house system has produced a ranking of all hospitals in the US and a focus list of states that we need to use in our analysis.

Your Python program should download the MS Excel Workbook from the following link:
http://kevincrook.com/utd/hospital_ranking_focus_states.xlsx

Your Python program should read this workbook. The first sheet is “**Hospitals National Ranking**” and contains a ranking list of all hospitals in the US, with columns “**Provider ID**” and “**Ranking**”. The second sheet is “**Focus States**” and contains a list of the focus states, with columns “**State Name**” and “**State Abbreviation**”.

Create the Hospital Ranking MS Excel Workbook

Your Python program should create a hospital ranking MS Excel Workbook named “**hospital_ranking.xlsx**” in the local directory without using any path names.

It should have a first sheet named “**Nationwide**”. It should have the following column headers “**Provider ID**”, “**Hospital Name**”, “**City**”, “**State**”, and “**County**”. Follow this header row with the top 100 hospitals as ranked by the in house proprietary system, ordered by rank. For the state column, the data should use the 2 letter state abbreviation.

For each of the states in the focus list, it should have a separate sheet for each state. The sheet name should be the state name spelled out, not an abbreviation. The sheets should be in alphabetic order by the state name spelled out. Each sheet should have the same columns and data as the first sheet, except the data should be the top 100 hospitals located in that state, ordered by rank.

Create the Measures Statistical Analysis MS Excel Workbook

Your Python program should create a hospital ranking MS Excel Workbook named "**measure_statistics.xlsx**" in the local directory without using any path names.

From the table **timely_and_effective_care___hospital** query out the **state**, **measure_id**, **measure_name**, and **score**. Some of the scores have non-numeric data, some have a mix of numeric and non-numeric data. If all scores for a measure are non-numeric, ignore that measure. If a score has a mix of numeric and non-numeric data, ignore the non-numeric data and just find statistics on the numeric data.

It should have a first sheet named "**Nationwide**". It should have the following column headers "**Measure ID**", "**Measure Name**", "**Minimum**", "**Maximum**", "**Average**", and "**Standard Deviation**". Follow this with 1 row per measure. Sort by measure_id. Calculate the minimum, maximum, average, and standard deviation for that measure for all hospitals nationwide.

Python Program

You will write a single file Python program. The program must download and read all files correctly. The program must run without stack trace. The program must create the specified output files.