30538 Problem Set 4

Yuliana Zhang and Dale Jin

2001-10-30

PS4: Due Sat Nov 2 at 5:00PM Central. Worth 100 points. We use (*) to indicate a problem that we think might be time consuming.

Style Points (10 pts)

Please refer to the minilesson on code style **here**.

Submission Steps (10 pts)

1. This problem set is a paired problem set.
2. Play paper, scissors, rock to determine who goes first. Call that person *Partner 1*.
    - Partner 1 (name and cnet ID):
    - Partner 2 (name and cnet ID):
3. Partner 1 will accept the `ps4` and then share the link it creates with their partner. You can only share it with one partner so you will not be able to change it after your partner has accepted.
4. "This submission is our work alone and complies with the 30538 integrity policy." Add your initials to indicate your agreement: **___** **___**
5. "I have uploaded the names of anyone else other than my partner and I worked with on the problem set **here**" (1 point)
6. Late coins used this pset: **___** Late coins left after submission: **___**
7. Knit your `ps4.qmd` to an PDF file to make `ps4.pdf`,
    - The PDF should not be more than 25 pages. Use `head()` and re-size figures when appropriate.
8. (Partner 1): push `ps4.qmd` and `ps4.pdf` to your github repo.
9. (Partner 1): submit `ps4.pdf` via Gradescope. Add your partner on Gradescope.
10. (Partner 1): tag your submission in Gradescope

Important: Repositories are for tracking code. **Do not commit the data or shapefiles to your repo.** The best way to do this is with `.gitignore`, which we have covered in class. If you do accidentally commit the data, Github has a guide. The best course of action depends on whether you have pushed yet. This also means that both partners will have to download the initial raw data and any data cleaning code will need to be re-run on both partners' computers.

Download and explore the Provider of Services (POS) file (10 pts)

```
import pandas as pd
import altair as alt

import warnings
warnings.filterwarnings('ignore')
```

1.

I pulled the following 7 variables:

PRVDR_CTGRY_SBTYP_CD (1 = short term), PRVDR_CTGRY_CD (1 = hospitals), PRVDR_NUM, PGM_TRMNTN_CD (00 = active provider), ZIP_CD, TRMNTN_EXPRTN_DT (not necessarily need), FAC_NAME.

2.

```
# Load 2016 CSV file
pos2016 = pd.read_csv("pos2016.csv")
pos2016['year'] = 2016

# Define a function to focus on short term hospitals
def filter_short_term_hospitals(data):
    return data[
        (data['PRVDR_CTGRY_SBTYP_CD'] == 1) &
        (data['PRVDR_CTGRY_CD'] == 1)
    ]

# Subset 2016
short_term_pos2016 = filter_short_term_hospitals(pos2016)
```

a.

```
# Count the number of short-term hospitals
hospitals_2016 = len(short_term_pos2016)
print(f'There are {hospitals_2016} short-term hospitals in 2016')

# Check whether there are closed hospitals
for cell in short_term_pos2016['PGM_TRMNTN_CD']:
    if cell == 1:
        print('This number of 2016 short-term hospitals does not make sense
        ↪  as it contains closed hospitals')
        break
```

```
There are 7245 short-term hospitals in 2016
This number of 2016 short-term hospitals does not make sense as it contains
closed hospitals
```

b. From the backgroud article provided in the assignment, "there are nearly 5,000 short-term, acute care hospitals in the United States", ref from "https://www.kff.org/report-section/a-look-at-rural-hospital-closures-and-implications-for-access-to-care-three-case-studies-issue-brief/". I think these numbers are different because the dataset contains a lot closed hospitals.

3.

```
# Load 2017 data
pos2017 = pd.read_csv("pos2017.csv")
pos2017['year'] = 2017
# Subset 2017
short_term_pos2017 = filter_short_term_hospitals(pos2017)
# Count the number of short-term hospitals
hospitals_2017 = len(short_term_pos2017)
print(f'There are {hospitals_2017} short-term hospitals in 2017')
# Check whether there are closed hospitals
for cell in short_term_pos2017['PGM_TRMNTN_CD']:
    if cell == 1:
        print('This number of 2017 short-term hospitals does not make sense
        ↪  as it contains closed hospitals')
        break

# Load 2018 data
pos2018 = pd.read_csv("pos2018.csv")
pos2018['year'] = 2018
```

```
# Subset 2018
short_term_pos2018 = filter_short_term_hospitals(pos2018)# Count the number
 ↪  of short-term hospitals
hospitals_2018 = len(short_term_pos2018)
print(f'There are {hospitals_2018} short-term hospitals in 2018')
# Check whether there are closed hospitals
for cell in short_term_pos2018['PGM_TRMNTN_CD']:
    if cell == 1:
        print('This number of 2018 short-term hospitals does not make sense
         ↪  as it contains closed hospitals')
        break

# Load 2019 data
pos2019 = pd.read_csv("pos2019.csv")
pos2019['year'] = 2019
# Subset 2019
short_term_pos2019 = filter_short_term_hospitals(pos2019)
# Count the number of short-term hospitals
hospitals_2019 = len(short_term_pos2019)
print(f'There are {hospitals_2019} short-term hospitals in 2019')
# Check whether there are closed hospitals
for cell in short_term_pos2019['PGM_TRMNTN_CD']:
    if cell == 1:
        print('This number of 2019 short-term hospitals does not make sense
         ↪  as it contains closed hospitals')
        break
```

```
There are 7260 short-term hospitals in 2017
This number of 2017 short-term hospitals does not make sense as it contains
closed hospitals
There are 7277 short-term hospitals in 2018
This number of 2018 short-term hospitals does not make sense as it contains
closed hospitals
There are 7303 short-term hospitals in 2019
This number of 2019 short-term hospitals does not make sense as it contains
closed hospitals
```

Same reasons as question 2.

```
# ref from
 ↪  https://www.askpython.com/python-modules/pandas/combine-csv-files-using-python
# Append 2016-2019 data
```

```
short_term_pos2016_2019 = pd.concat([short_term_pos2016, short_term_pos2017,
 ↪  short_term_pos2018, short_term_pos2019], ignore_index=True)

# Build the overall dataframe
pos = short_term_pos2016_2019

# Plot the number of observations by year.
## Rebuild data frame for chart
observations = pos.groupby(['year']).size().reset_index(name = 'count')

chart = alt.Chart(observations, title = 'The Number of Observations by
 ↪  Year').mark_line().encode(
    alt.X('year:O', title = 'Year'),
    alt.Y('count:Q', title = 'Number of Observations', scale =
 ↪  alt.Scale(domainMin = 7000)),
).properties(
    width = 200,
    height = 100
)
chart.show()
```

```
alt.Chart(...)
```

4.a

```
# Filter unique hospitals by dropping duplicates based on 'CMS_ID' and 'Year'
unique_hospitals = pos.drop_duplicates(subset = ['PRVDR_NUM', 'year'])

# Count unique hospitals per year
unique_hospitals =
 ↪  unique_hospitals.groupby('year')['PRVDR_NUM'].size().reset_index(name =
 ↪  'unique_hospitals')

# Plot the number of unique CMS by year.
chart = alt.Chart(unique_hospitals, title = 'The Number of Unique Hospitals
 ↪  by Year').mark_line().encode(
    alt.X('year:O', title = 'Year'),
    alt.Y('unique_hospitals:Q', title = 'Number of Unique Hospitals', scale =
 ↪  alt.Scale(domainMin = 7000)),
).properties(
    width = 200,
```

```
    height = 100
)
chart.show()
```

```
alt.Chart(...)
```

4.b

These two graphs are almost the same, which shows that the dataset does not contain repeated hospitals in each year. Thus, the reason for question 2 are solely based on the closed hospitals.

Identify hospital closures in POS file (15 pts) (*)

1.
2.
3. a.
   b.
   c.

Download Census zip code shapefile (10 pt)

1. a.
   b.
2.

Calculate zip code's distance to the nearest hospital (20 pts) (*)

1.
2.
3.
4. a.
   b.
5. a.
   b.
   c.

Effects of closures on access in Texas (15 pts)

1.
2.
3.
4.

Reflecting on the exercise (10 pts)