Problem Set 4

PS4: Due Sat Nov 2 at 5:00PM Central. Worth 100 points.

Style Points (10 pts)

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 (Ella Montgomery; emontgomery2): • Partner 2 (Mitch Bobbin; mbobbin):
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: **EM MB**
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: **1** Late coins left after submission: **2**

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

1. We chose to include the following variables:

- Hospital Type: **PRVDR_CTGRY_SBTYP_CD**, **PRVDR_CTGRY_CD**
- CMS Certification number: **PRVDR_NUM**
- Facility name: **FAC_NAME**
- Termination code: **PGM_TRMNTN_CD**
- ZIP Code: **ZIP_CD**
- State code: **STATE_CD**

2. a.

::: {.cell execution_count=1} “‘ {.python .cell-code} import pandas as pd import os #only need to change base path when we’re switching off working base_path=r”C:\Users\EM\Documents\GitHub\problem-set-4-mitchella”

file_path2016=r”pos2016.csv”

path2016=os.path.join(base_path,file_path2016)

#import the data for 2016 and store in a df df_pos2016=pd.read_csv(path2016)

df_pos2016.shape df_pos2016.groupby(“PRVDR_CTGRY_SBTYP_CD”).count()

#filter to the correct provider category and subcategory:

df_pos2016=df_pos2016[(df_pos2016[“PRVDR_CTGRY_SBTYP_CD”]==1) & (df_pos2016[“PRVDR_CTGRY_CD”] == 1)]

df_pos2016.shape

#take the number of rows from the filtered dataset over the #number of rows in the unfiltered to get the proportion that #are short-term hospitals:

7245/141557 “‘

::: {.cell-output .cell-output-display execution_count=1} `0.0511807964282939` ::: :::

There are 7,245 short term hospitals in this data. This represents about 5% of all providers in the data.

b.
This number may not make sense. This is a huge overestimate of the Kaiser Family Foundation's figure of around 5,000 "short-term acute hospitals", which they also published in 2016. It could differ because the two sources classify short term hospitals differently, with the Kaiser Family Foundation having a stricter definition than Medicare and Medicaid.

3.

```
file_path2017=r"pos2017.csv"

path2017=os.path.join(base_path,file_path2017)

#import the data for 2016 and store in a df
df_pos2017=pd.read_csv(path2017)

#filter to the correct provider category and subcategory:
```

```
df_pos2017=df_pos2017[(
  df_pos2017["PRVDR_CTGRY_SBTYP_CD"]==1) & (
    df_pos2017["PRVDR_CTGRY_CD"] == 1)]

df_pos2017.shape
```

```
(7260, 7)
```

7,260 short term hospitals in 2017.

```
file_path2018=r"pos2018.csv"

path2018=os.path.join(base_path,file_path2018)

#import the data for 2018 and store in a df
#for some reason we needed to specify encoding. used chatgpt
#to troubleshoot when it wouldnt load in.
df_pos2018=pd.read_csv(path2018,encoding="latin1")

#@Ella I think you and I have different names for the #subgroup column when
 ↪  we've exported. for some reason #mine is loading this way
df_pos2018=df_pos2018[(
  df_pos2018["PRVDR_CTGRY_SBTYP_CD"]==1) & (
    df_pos2018["PRVDR_CTGRY_CD"] == 1)]

df_pos2018.shape
```

```
(7277, 7)
```

7,277 hospitals in 2018.

```
file_path2019=r"pos2019.csv"

path2019=os.path.join(base_path,file_path2019)

#import the data for 2019 and store in a df
#for some reason we needed to specify encoding. used
#chatgpt to troubleshoot when it wouldnt load in.

df_pos2019=pd.read_csv(path2019,encoding="latin1")

df_pos2019=df_pos2019[(
```

```
  df_pos2019["PRVDR_CTGRY_SBTYP_CD"]==1) & (
    df_pos2019["PRVDR_CTGRY_CD"] == 1)]

df_pos2019.shape
```

```
(7303, 7)
```

7,303 hospitals in 2019.

```
#add each dfs associated year to every observation:
df_pos2016["year"]=2016
df_pos2017["year"]=2017
df_pos2018["year"]=2018
df_pos2019["year"]=2019
#combine the dfs
combined_df_pos=pd.concat([df_pos2016,df_pos2017,df_pos2018,df_pos2019])

#plot the combined df
import altair as alt
alt.data_transformers.enable("vegafusion")

observation_plot=alt.Chart(combined_df_pos).mark_bar().encode(
  alt.X("year:O"),
  alt.Y("count()")
)
```

4. a.

```
#gives the number of unique values for PRVDR_NUM by year:
print(combined_df_pos.groupby("year")["PRVDR_NUM"].nunique())

#plot
alt.data_transformers.enable("vegafusion")

hospital_plot=alt.Chart(combined_df_pos).mark_bar().encode(
  alt.X("year:O"),
  alt.Y("distinct(PRVDR_NUM):Q")
)

#display both plots side by side
hospital_plot | observation_plot
```

```
year
2016    7245
2017    7260
2018    7277
2019    7303
Name: PRVDR_NUM, dtype: int64
```

```
alt.HConcatChart(...)
```

b.
The plots show us that each individual row is a single hospital, because the plots are identical. In the first plot, we examined the number of rows for each year, and in the second, we examined the number of unique values for PRVDR_NUM, and they appear to be identical.

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

1.

```
#active in 2016
active_2016 = combined_df_pos[(combined_df_pos['year'] == 2016) &
 ↪  (combined_df_pos['PGM_TRMNTN_CD'] == 0)]

#compare against following years
merged = active_2016[['PRVDR_NUM', 'FAC_NAME', 'ZIP_CD']].merge(
    combined_df_pos,
    on='PRVDR_NUM',
    how='left'
)

#filter out active hospitals
closed_hospitals = merged[(merged['year'] > 2016) & (merged['PGM_TRMNTN_CD']
 ↪  == 1)
]

print(closed_hospitals['PRVDR_NUM'].nunique())
```

```
133
```

There are 133 hospitals that fit this suspected closure definition.

2.

```
print(closed_hospitals[['FAC_NAME_x',
 ↪  'year']].sort_values(by='FAC_NAME_x').head(10))
```

```
                                      FAC_NAME_x  year
13317                       ABRAZO MARYVALE CAMPUS  2017
13318                       ABRAZO MARYVALE CAMPUS  2018
13319                       ABRAZO MARYVALE CAMPUS  2019
12493      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  2017
12494      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  2018
12495      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  2019
4587                      AFFINITY MEDICAL CENTER  2019
4586                      AFFINITY MEDICAL CENTER  2018
5613   ALBANY MEDICAL CENTER / SOUTH CLINICAL CAMPUS  2017
5614   ALBANY MEDICAL CENTER / SOUTH CLINICAL CAMPUS  2018
```

3.

```
closed_hospitals = closed_hospitals[['PRVDR_NUM', 'FAC_NAME_x', 'ZIP_CD_x',
 ↪  'PRVDR_CTGRY_SBTYP_CD', 'PRVDR_CTGRY_CD', 'STATE_CD', 'PGM_TRMNTN_CD',
 ↪  'year']]

closed_hospitals.rename(columns={'FAC_NAME_x': 'FAC_NAME', 'ZIP_CD_x':
 ↪  'ZIP_CD'}, inplace=True)

#yearly count of active hospitals by ZIP code
yearly_zip_active_pos = combined_df_pos[combined_df_pos['PGM_TRMNTN_CD'] ==
 ↪  0].groupby(['year', 'ZIP_CD']).size().reset_index(name='active_pos')


#do we need to do the same test with looking at from #16-17?


all_years = yearly_zip_active_pos['year'].unique()
all_zip_codes = yearly_zip_active_pos['ZIP_CD'].unique()
all_combinations = pd.MultiIndex.from_product([all_years, all_zip_codes],
 ↪  names=['year', 'ZIP_CD']).to_frame(index=False)

#Merge with the existing data to include missing #combinations
complete_yearly_zip_active_pos =
 ↪  all_combinations.merge(yearly_zip_active_pos, on=['year', 'ZIP_CD'],
 ↪  how='left')
```

```
#Fill missing 'active_pos' values with 0 for cases #without observations
complete_yearly_zip_active_pos['active_pos'] =
 ↪  complete_yearly_zip_active_pos['active_pos'].fillna(0)

#identify zip codes where number of hospitals is changing

zip_changes =
 ↪  complete_yearly_zip_active_pos.groupby("ZIP_CD")['active_pos'].nunique().reset_index()
complete_yearly_zip_active_pos =
 ↪  complete_yearly_zip_active_pos.sort_values(['ZIP_CD', 'year'])

#Calculate the difference in active_pos by year within #each ZIP code
complete_yearly_zip_active_pos['active_pos_diff'] =
 ↪  complete_yearly_zip_active_pos.groupby('ZIP_CD')['active_pos'].diff()

#Identify ZIP codes with any decrease in active_pos over #time.
#If there's a zip code with a decrease, we can assume the #closure is "real"
 ↪  and not due to a M&A
zip_codes_with_decrease =
 ↪  complete_yearly_zip_active_pos[complete_yearly_zip_active_pos['active_pos_diff']
 ↪  < 0]['ZIP_CD'].unique()

#Filter the closed_hospitals DataFrame for those in ZIP #codes with a
 ↪  decrease in active_pos
actual_closures =
 ↪  closed_hospitals[closed_hospitals['ZIP_CD'].isin(zip_codes_with_decrease)]

len(actual_closures["PRVDR_NUM"].unique())

#130 of the 133 are actual closures. That means 3 are M&As
```

130

a. There are 3 potential mergers.
b. 130 hospitals are suspected to be an actual closure, due to our methodology of filtering out any hospital that was in a zip code where they didn't experience a decrease in the number of hospitals.

c.

```
print(actual_closures.sort_values("FAC_NAME").head(10))
```

```
      PRVDR_NUM                                    FAC_NAME   ZIP_CD  \
13317    030001                      ABRAZO MARYVALE CAMPUS  85031.0
13318    030001                      ABRAZO MARYVALE CAMPUS  85031.0
13319    030001                      ABRAZO MARYVALE CAMPUS  85031.0
12495    050196      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  93230.0
12493    050196      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  93230.0
12494    050196      ADVENTIST MEDICAL CENTER - CENTRAL VALLEY  93230.0
4586     360151                     AFFINITY MEDICAL CENTER  44646.0
4587     360151                     AFFINITY MEDICAL CENTER  44646.0
5613     330189  ALBANY MEDICAL CENTER / SOUTH CLINICAL CAMPUS  12208.0
5614     330189  ALBANY MEDICAL CENTER / SOUTH CLINICAL CAMPUS  12208.0

       PRVDR_CTGRY_SBTYP_CD  PRVDR_CTGRY_CD STATE_CD  PGM_TRMNTN_CD  year
13317                   1.0               1       AZ              1  2017
13318                   1.0               1       AZ              1  2018
13319                   1.0               1       AZ              1  2019
12495                   1.0               1       CA              1  2019
12493                   1.0               1       CA              1  2017
12494                   1.0               1       CA              1  2018
4586                    1.0               1       OH              1  2018
4587                    1.0               1       OH              1  2019
5613                    1.0               1       NY              1  2017
5614                    1.0               1       NY              1  2018
```

Download Census zip code shapefile (10 pt)

1. a. The five file types are as follows:

   - .dbf, which contains attribute informaion in a table format
   - .prj, which contains information about units and the Coordinate Reference System (CRS)
   - .shp, which contains feature geometry
   - .shx, which contains the positional index information
   - .xml, which is written in a markup language and contains plaintext metadata and plainkeys

   b.

   - .dbf: 6275 kb
   - .prj: 1 kb

   - .shp: 817915 kb
   - .shx: 259 kb

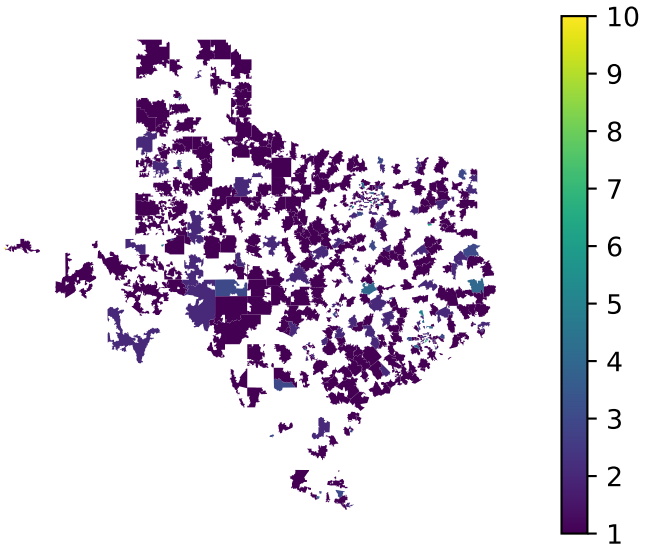
- .xml: 16 kb

2.

```
import geopandas as gpd

#shp_path =
 ↪  r"C:\\Users\\Mitch\\Documents\\GitHub\\problem-set-4-mitchella\\gz_2010_us_860_00_500k.z

#@Ella: I had to change the path a bit to get this to #run. Adjust as needed
 ↪  for your own purposes yours was:
shp_path =
 ↪  'C:\\Users\\EM\\Documents\\GitHub\\problem-set-4-mitchella\\gz_2010_us_860_00_500k\\gz_2

geo_data = gpd.read_file(shp_path)

#TX ZIPs start with 75-79
tx_geo_data = geo_data[geo_data['ZCTA5'].astype(str).str.startswith(('75',
 ↪  '76', '77', '78', '79'))]
```

```
import matplotlib.pyplot as plt

#group by zipcode and count number of unique POS numbers for each group
pos_per_zip =
 ↪  df_pos2016.groupby('ZIP_CD')['PRVDR_NUM'].nunique().reset_index()
pos_per_zip.columns = ['ZCTA5', '# of POS']

#convert ZIP from float to string
pos_per_zip['ZCTA5'] = pos_per_zip['ZCTA5'].astype(int)
pos_per_zip['ZCTA5'] = pos_per_zip['ZCTA5'].astype(str).str.zfill(5)

tx_pos_per_zip = pos_per_zip[pos_per_zip['ZCTA5'].str.startswith(('75', '76',
 ↪  '77', '78', '79'))]

#format zip codes
tx_geo_data['ZCTA5'] = tx_geo_data['ZCTA5'].astype(str).str.zfill(5)

tx_geo_data_1 = tx_geo_data.merge(tx_pos_per_zip, on='ZCTA5', how='left')

tx_geo_data_1.plot(column = '# of POS',
legend = True).set_axis_off()
```

```
C:\Users\EM\anaconda3\Lib\site-packages\geopandas\geodataframe.py:1819:
SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus
```



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

1.

```
zips_all_centroids = geo_data.assign(centroid=geo_data.centroid)
zips_all_centroids.shape
```

```
C:\Users\EM\AppData\Local\Temp\ipykernel_29140\3746452380.py:1: UserWarning:

Geometry is in a geographic CRS. Results from 'centroid' are likely
incorrect. Use 'GeoSeries.to_crs()' to re-project geometries to a projected
CRS before this operation.


(33120, 7)
```

The resulting GeoDataFrame is 33120 x 7. The columns are taken from census documentation: - GEO_ID: Concatenation of 2010 State and County - ZCTA5: ZIP Code Tabulation Area - NAME: ZIP Code - LSAD: legal/statistical area description - CENSUSAREA: calculated area derived from the ungeneralized area of each ZIP Code - geometry: polygonal shape of designated area - centroid: center point of designated area

2.

```
zips_texas_centroids =
 ↪  zips_all_centroids[zips_all_centroids['ZCTA5'].str.startswith(('75',
 ↪  '76', '77', '78', '79'))]

border_ranges = ( '870', '871', '872', '873', '874', '875', '876', '877',
 ↪  '878', '879', '880', '881', '882', '883', '884',  #New Mexico
'73', '74',  #Oklahoma
'716', '717', '718', '719', '720', '721', '722', '723', '724', '725', '726',
 ↪  '727', '728', '729',  #Arkansas
'700', '701', '702', '703', '704', '705', '706', '707', '708', '709', '710',
 ↪  '711', '712', '713', '714', '715',  #Louisiana
'75', '76', '77', '78', '79'
)

zips_texas_borderstates_centroids =
 ↪  zips_all_centroids[zips_all_centroids['ZCTA5'].str.startswith((border_ranges))]

print(zips_texas_centroids['ZCTA5'].nunique())
print(zips_texas_borderstates_centroids['ZCTA5'].nunique())
```

```
1935
4057
```

The subset zips_texas_centroids has 1,935 unique values, while the borderstates subset has 4057 unique entries. The latter subset includes Texan ZIP Codes as well.

3.

```
zips_with_hospitals = pos_per_zip[pos_per_zip['# of POS'] > 0]
zips_withhospital_centroids =
 ↪  zips_texas_borderstates_centroids.merge(zips_with_hospitals, on='ZCTA5',
 ↪  how='inner')
```

For this case, an inner merge on the ZIP Code variable was used to exclusively retain ZIP Codes that have more than one hospital.