\usepackage{fvextra} \DefineVerbatimEnvironment{Highlighting}{Verbatim}{breaklines,commandchars=\\\{\}}

\RecustomVerbatimEnvironment{verbatim}{Verbatim}{ showspaces = false, showtabs = false, breaksymbolleft={}, breaklines }

Your Title

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)

1. What are the Variables I pulled?

```
import pandas as pd
import altair as alt
import os
import geopandas as gpd
import warnings
warnings.filterwarnings("ignore")
```

```
# Reading dataset
path = "/Users/tsaili-ting/Uchicago/Year2/Y2Fall/Python2/problem-set-4-gena-ting"

path_2016 = os.path.join(path, "pos2016.csv")
pos2016 = pd.read_csv(path_2016)

#variable pull

pos2016.columns
```

```
Index(['PRVDR_CTGRY_SBTYP_CD', 'PRVDR_CTGRY_CD', 'CITY_NAME', 'FAC_NAME',
       'PRVDR_NUM', 'PGM_TRMNTN_CD', 'TRMNTN_EXPRTN_DT', 'ZIP_CD'],
      dtype='object')
```

1. Provider Category , 2. Provider Category sub , 3. City, 4. Hospital Name, 5. CMS Number, 6. Termination code, 7. Termination or Expiration Date, 8. ZIP code

2. a. How many hospitals are reported in this data?

```
pos2016_s = pos2016.loc[(pos2016["PRVDR_CTGRY_CD"]==1)&(pos2016["PRVDR_CTGRY_SBTYP_CD"]==

len(pos2016_s)
```

```
7245
```

```
The number of short-term hospital in the dataset is 7245.

b.Does this number make sense? Cross-reference with other sources and cite the numberyou compared it to.

 However, according to American Hospital Association(https://www.aha.org/statistics/fast-facts-us-hospitals), there are only 5129 community hospital. Citing that "Excluded are hospitals not accessible by the general public, such as prison hospitals or college infirmaries." in AHA's number.
```

3. Plot the number of observations in your dataset by year.

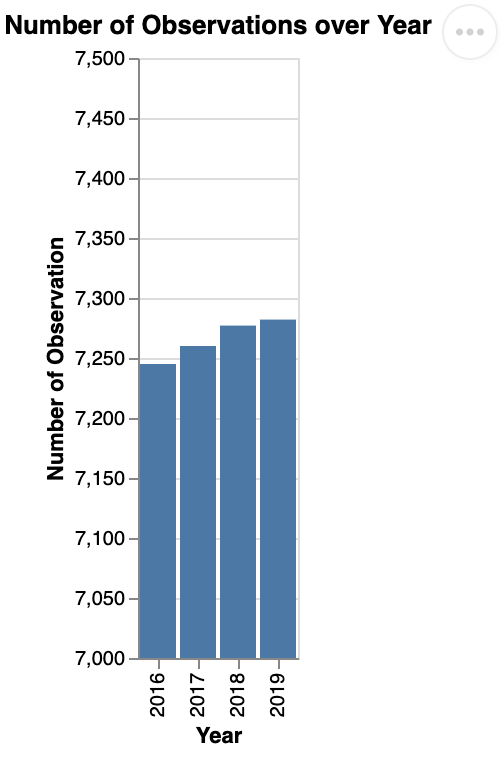
```
# Read 2017,18,19 dataset
path_2017 = os.path.join(path, "pos2017.csv")
path_2018 = os.path.join(path, "pos2018.csv")
path_2019 = os.path.join(path, "pos2019.csv")
```

```
pos2017 = pd.read_csv(path_2017)
pos2018 = pd.read_csv(path_2018,encoding="latin1")
pos2019 = pd.read_csv(path_2019,encoding="latin1")
```

```
# filter only short term hospital
pos2017_s = pos2017.loc[(pos2017["PRVDR_CTGRY_CD"]==1)&(pos2017["PRVDR_CTGRY_SBTYP_CD"]==
pos2018_s = pos2018.loc[(pos2018["PRVDR_CTGRY_CD"]==1)&(pos2018["PRVDR_CTGRY_SBTYP_CD"]==
pos2019_s = pos2019.loc[(pos2019["PRVDR_CTGRY_CD"]==1)&(pos2019["PRVDR_CTGRY_SBTYP_CD"]==
```

```
# Add a column represent the fisical year, and append data

pos2016_s["Year"] = "2016"
pos2017_s["Year"] = "2017"
pos2018_s["Year"] = "2018"
pos2019_s["Year"] = "2019"

# Append the dataset
pos_combine = pd.concat([pos2016_s, pos2017_s, pos2018_s, pos2019_s], ignore_index=True)
```

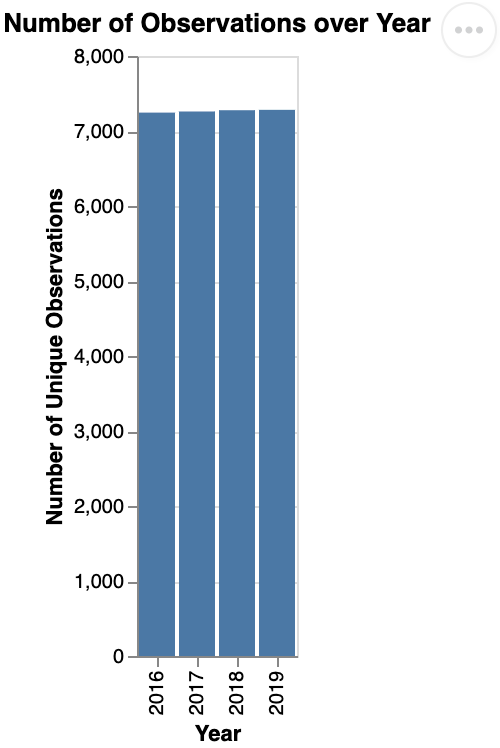
```
# plot the observations by year
alt.data_transformers.disable_max_rows()
alt.Chart(pos_combine).mark_bar().encode(
    x=alt.X('Year:N',
            title="Year"),
    y=alt.Y('count()', title="Number of Observation",scale=alt.Scale(domain=[7000, 7500])
).properties( title='Number of Observations over Year')
```



4. a. Plot the number of unique hospitals in your dataset per year.

```
# aggregate by year and calculate the unique number
unique_hospital = pos_combine.groupby('Year')['PRVDR_NUM'].nunique().reset_index()
```

```
alt.Chart(unique_hospital).mark_bar().encode(
    x=alt.X('Year:N', title="Year"),
    y=alt.Y('PRVDR_NUM:Q', title="Number of Unique Observations")
    ).properties(
    title='Number of Observations over Year'
)
```



b.

```
pos_combine.groupby("Year").agg(NumberOfHospital = ("Year","count")).reset_index()
```

	Year	NumberOfHospital
0	2016	7245
1	2017	7260
2	2018	7277
3	2019	7282

```
unique_hospital
```

	Year	PRVDR_NUM
0	2016	7245

	Year	PRVDR_NUM
1	2017	7260
2	2018	7277
3	2019	7282

They are exactly the same number, meaning each row is a unique hospital.

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

1.

```
years = ["2016", "2017", "2018", "2019"]

data = {}
for year in years:
    file_path = os.path.join(path, f"pos{year}.csv")

    df = pd.read_csv(file_path, encoding='ISO-8859-1')
    data[year] = df[(df["PRVDR_CTGRY_CD"] == 1) & (df["PRVDR_CTGRY_SBTYP_CD"] == 1)]

closures = []
for i, year in enumerate(years[:-1]):
    current_year_data = data[year]
    next_year_data = data[years[i + 1]]


    merged_data = current_year_data.merge(
        next_year_data[['PRVDR_NUM']],
        on='PRVDR_NUM',
        how='left',
        indicator=True
    )

    closed_hospitals = merged_data[
        (merged_data['_merge'] == 'left_only') |
        (merged_data['TRMNTN_EXPRTN_DT'].notna())
    ]

    closures.extend(
        closed_hospitals[['FAC_NAME', 'ZIP_CD', 'PRVDR_NUM']].assign(Year_Closed=years[i
    )

closures_df = pd.DataFrame(closures)
print(f"Total suspected closures: {len(closures_df)}")
print(closures_df)
```

```
Total suspected closures: 11606
                                                FAC_NAME   ZIP_CD PRVDR_NUM  \
```

```
0                              NORTH JACKSON HOSPITAL  35740.0    010004
1                            HARTSELLE MEDICAL CENTER  35640.0    010009
2                       MARSHALL MEDICAL CENTER NORTH  35976.0    010010
3                     SOUTHWEST ALABAMA MEDICAL CENTER  36784.0    010015
4                                FAIRVIEW MEDICAL CTR  36108.0    010017
...                                               ...      ...       ...
11601       SELECT SPECIALTY HOSPITAL DALLAS DOWNTOWN  75246.0    670100
11602      SELECT SPECIALTY HOSPITAL DALLAS (GARLAND)  75042.0    670101
11603              VICTORY MEDICAL CENTER BEAUMONT LP  77706.0    670104
11604               HERMAN DRIVE SURGICAL HOSPITAL LP  77004.0    670105
11605  CONTINUECARE HOSPITAL AT MEDICAL CENTER ODESSA  79761.0    670111

      Year_Closed
0            2017
1            2017
2            2017
3            2017
4            2017
...           ...
11601        2019
11602        2019
11603        2019
11604        2019
11605        2019

[11606 rows x 4 columns]
```

The total suspected closure amount is 11606 hospitals. 2.

```
sorted_closures = closures_df.sort_values(by='FAC_NAME').head(10)
sorted_closures[['FAC_NAME', 'Year_Closed']]
print(sorted_closures[['FAC_NAME', 'Year_Closed']])
```

```
                                        FAC_NAME Year_Closed
3002     (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL        2017
6849     (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL        2018
10756    (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL        2019
6927             (CLOSED) BAPTIST HOSPITAL FOR WOMEN        2018
10838            (CLOSED) BAPTIST HOSPITAL FOR WOMEN        2019
6851       (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY        2018
3004       (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY        2017
10758      (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY        2019
3030   (CLOSED) BAPTIST MEMORIAL HOSPITAL LAUDERDALE        2017
6877   (CLOSED) BAPTIST MEMORIAL HOSPITAL LAUDERDALE        2018

                                        FAC_NAME Year_Closed
```

3002 (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL 2017 6849 (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL 2018 10756 (CLOSED) BAPTIST HICKMAN COMMUNITY HOSPITAL 2019 6927 (CLOSED) BAPTIST HOSPITAL FOR WOMEN 2018 10838 (CLOSED) BAPTIST HOSPITAL FOR WOMEN

2019 6851 (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY 2018 3004 (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY 2017 10758 (CLOSED) BAPTIST HOSPITAL OF ROANE COUNTY 2019 3030 (CLOSED) BAPTIST MEMORIAL HOSPITAL LAUDERDALE 2017 6877 (CLOSED) BAPTIST MEMORIAL HOSPITAL LAUDERDALE 2018

3. a.

```
active_hospitals_by_zip = {}
for year, df in data.items():

    active_by_zip = df[df['TRMNTN_EXPRTN_DT'].isna()].groupby('ZIP_CD')['PRVDR_NUM'].coun
    active_hospitals_by_zip[year] = active_by_zip

potential_mergers = []
for _, row in closures_df.iterrows():
    zip_code = row['ZIP_CD']
    closure_year = row['Year_Closed']

    closure_year_str = str(closure_year)
    next_year_str = str(int(closure_year) + 1)


    if closure_year_str in active_hospitals_by_zip and next_year_str in active_hospitals_

        current_count = active_hospitals_by_zip[closure_year_str].get(zip_code, 0)
        next_count = active_hospitals_by_zip[next_year_str].get(zip_code, 0)


        if current_count == next_count:
            potential_mergers.append(row)


potential_mergers_df = pd.DataFrame(potential_mergers)


print(f"Number of suspected closures that may be mergers/acquisitions: {len(potential_mer
print(potential_mergers_df[['FAC_NAME', 'ZIP_CD', 'Year_Closed']])
```

```
Number of suspected closures that may be mergers/acquisitions: 7625
                                          FAC_NAME   ZIP_CD Year_Closed
0                            NORTH JACKSON HOSPITAL  35740.0        2017
1                          HARTSELLE MEDICAL CENTER  35640.0        2017
2                     MARSHALL MEDICAL CENTER NORTH  35976.0        2017
3                   SOUTHWEST ALABAMA MEDICAL CENTER  36784.0        2017
4                              FAIRVIEW MEDICAL CTR  36108.0        2017
...                                             ...      ...         ...
7678       SELECT SPECIALTY HOSPITAL DALLAS DOWNTOWN  75246.0        2018
7679      SELECT SPECIALTY HOSPITAL DALLAS (GARLAND)  75042.0        2018
7680              VICTORY MEDICAL CENTER BEAUMONT LP  77706.0        2018
7681              VICTORY MEDICAL CENTER HOUSTON, LP  77004.0        2018
```

```
7682  CONTINUECARE HOSPITAL AT MEDICAL CENTER ODESSA  79761.0        2018

[7625 rows x 3 columns]
```

Number of suspected closures that may be mergers/acquisitions: 7625 b.

```python
active_hospitals_by_zip = {}
for year, df in data.items():

    active_by_zip = df[df['TRMNTN_EXPRTN_DT'].isna()].groupby('ZIP_CD')['PRVDR_NUM'].coun
    active_hospitals_by_zip[year] = active_by_zip


filtered_closures = []
for _, row in closures_df.iterrows():
    zip_code = row['ZIP_CD']
    closure_year = row['Year_Closed']


    closure_year_str = str(closure_year)
    next_year_str = str(int(closure_year) + 1)


    if closure_year_str in active_hospitals_by_zip and next_year_str in active_hospitals_

        current_count = active_hospitals_by_zip[closure_year_str].get(zip_code, 0)
        next_count = active_hospitals_by_zip[next_year_str].get(zip_code, 0)


        if current_count > next_count:
            filtered_closures.append(row)

filtered_closures_df = pd.DataFrame(filtered_closures)
print(f"Total filtered suspected closures: {len(filtered_closures_df)}")
print(filtered_closures_df[['FAC_NAME', 'ZIP_CD', 'Year_Closed']])
```

```
Total filtered suspected closures: 42
                                        FAC_NAME    ZIP_CD Year_Closed
125                 FLORENCE COMMUNITY HEALTHCARE  85132.0        2017
252           LONG BEACH COMMUNITY MEDICAL CENTER  90804.0        2017
575                          ST MARYS HOSPITAL INC  33407.0        2017
594              METROPOLITAN HOSPITAL OF MIAMI  33126.0        2017
769              SEMPERCARE HOSPITAL OF AUGUSTA  30901.0        2017
906                           ST MARY HOSPITAL INC  62301.0        2017
937                              MEMORIAL HOSPITAL  62233.0        2017
946                         ILLINOIS VETERANS HOME  62301.0        2017
1340                        BOSSIER MEDICAL CENTER  71111.0        2017
1434                    BOSSIER SPECIALTY HOSPITAL  71111.0        2017
1830                      HANCOCK MEDICAL HOSPITAL  39521.0        2017
1901                        MCCUNE BROOKS HOSPITAL  64836.0        2017
2143                            METHODIST HOSPITAL  68131.0        2017
```

2172	CONTINUECARE HOSPITAL OF CARSON TAHOE INC	89703.0	2017
2334	LAKE SHORE HOSPITAL	14081.0	2017
2419	JOHN UMSTEAD HOSPITAL	27509.0	2017
2523	AFFINITY MEDICAL CENTER - MASSILLON CAMPUS	44646.0	2017
2678	CITY OF FAITH HOSPITAL	74137.0	2017
3044	(CLOSED) TRINITY HOSPITAL	37061.0	2017
3214	VISTA HOSPITAL OF DALLAS	75042.0	2017
3455	SEMPERCARE HOSPITAL OF LONGVIEW	75601.0	2017
3759	SELECT SPECIALTY HOSPITAL FOX VALLEY	54904.0	2017
3819	SELECT SPECIALTY HOSPITAL DALLAS (GARLAND)	75042.0	2017
4034	MT ZION HOSP & MED CTR OF THE UCSF	94115.0	2018
4123	PACIFIC COAST HOSPITAL	94115.0	2018
5218	WOMEN'S AND CHILDREN'S HOSPITAL	70508.0	2018
5271	SOUTHPARK COMMUNITY HOSPITAL	70508.0	2018
5314	PROVIDENT HOSP	21215.0	2018
5321	LIBERTY MED CENTER	21215.0	2018
5657	LAIRD HOSPITAL	39365.0	2018
6528	AMERICAN TRANSITIONAL HOSPITAL	73112.0	2018
6935	LANDMARK MEDICAL CENTER	79902.0	2018
6938	SOUTHWESTERN GENERAL HOSPITAL	79902.0	2018
6948	ST EDWARD HOSPITAL	76520.0	2018
6979	MILAM REGIONAL MEDICAL CENTER	76520.0	2018
6992	DALLAS/FORT WORTH MEDICAL CENTER	75051.0	2018
7081	HAMILTON GENERAL HOSPITAL	76531.0	2018
7239	LANDMARK MED CTR	79902.0	2018
7245	SOUTHWESTERN GENERAL HOSPITAL	79902.0	2018
7253	CENTRAL TEXAS HOSPITAL	76520.0	2018
7301	SCCI HOSPITAL EL PASO	79902.0	2018
7651	EL PASO LTAC HOSPITAL	79902.0	2018

Total filtered suspected closures: 42

c.

```
sorted_filtered_closures = filtered_closures_df.sort_values(by='FAC_NAME').head(10)
print(sorted_filtered_closures[['FAC_NAME', 'ZIP_CD', 'Year_Closed']])
```

	FAC_NAME	ZIP_CD	Year_Closed
3044	(CLOSED) TRINITY HOSPITAL	37061.0	2017
2523	AFFINITY MEDICAL CENTER - MASSILLON CAMPUS	44646.0	2017
6528	AMERICAN TRANSITIONAL HOSPITAL	73112.0	2018
1340	BOSSIER MEDICAL CENTER	71111.0	2017
1434	BOSSIER SPECIALTY HOSPITAL	71111.0	2017
7253	CENTRAL TEXAS HOSPITAL	76520.0	2018
2678	CITY OF FAITH HOSPITAL	74137.0	2017
2172	CONTINUECARE HOSPITAL OF CARSON TAHOE INC	89703.0	2017
6992	DALLAS/FORT WORTH MEDICAL CENTER	75051.0	2018
7651	EL PASO LTAC HOSPITAL	79902.0	2018

	FAC_NAME	ZIP_CD	Year_Closed

3044 (CLOSED) TRINITY HOSPITAL 37061.0 2017 2523 AFFINITY MEDICAL CENTER - MASSILLON CAMPUS 44646.0 2017 6528 AMERICAN TRANSITIONAL HOSPITAL 73112.0 2018 1340 BOSSIER MEDICAL CENTER 71111.0 2017 1434 BOSSIER SPECIALTY HOSPITAL 71111.0 2017 7253 CENTRAL TEXAS HOSPITAL 76520.0 2018 2678 CITY OF FAITH HOSPITAL 74137.0 2017 2172 CONTINUECARE HOSPITAL OF CARSON TAHOE INC 89703.0 2017 6992 DALLAS/FORT WORTH MEDICAL CENTER 75051.0 2018 7651 EL PASO LTAC HOSPITAL 79902.0 2018

Download Census zip code shapefile (10 pt)

1. a.

   - gz_2010_us_860_00_500k.dbf: It's an Attribute Table. This file contains attribute data for each shape (e.g., names, population, or other data related to each spatial feature). It's essentially a database in a table format and is essential for linking spatial features to their descriptive information.
   - gz_2010_us_860_00_500k.shp: It's a Shapefile. It contains the geometry data or spatial information. such as the points, lines, or polygons that make up the shape of each feature in the dataset.
   - gz_2010_us_860_00_500k.xml: It's a Metadata File. This file includes metadata, such as a description of the data, its source, creation date, and other details about the dataset.
   - gz_2010_us_860_00_500k.prj: It's a Projection File. This file holds information about the coordinate system and projection. It ensures that the shapefile aligns properly on a map with other spatial data.
   - gz_2010_us_860_00_500k.shx: It's a Shape Index File. This index file is used to facilitate quick access to the geometries in the .shp file. It provides the spatial index of the geometry, helping software read specific features quickly without searching through the entire .shp file.

   b.
   - gz_2010_us_860_00_500k.dbf: 6.4MB
   - gz_2010_us_860_00_500k.shp: 837.5MB
   - gz_2010_us_860_00_500k.xml: 16KM
   - gz_2010_us_860_00_500k.prj: 165bytes
   - gz_2010_us_860_00_500k.shx: 265KB
2.

```
# read the shapefile
filepath = "/Users/tsaili-ting/Uchicago/Year2/Y2Fall/Python2/problem-set-4-gena-ting/gz_2
data = gpd.read_file(filepath)
type(data)
```

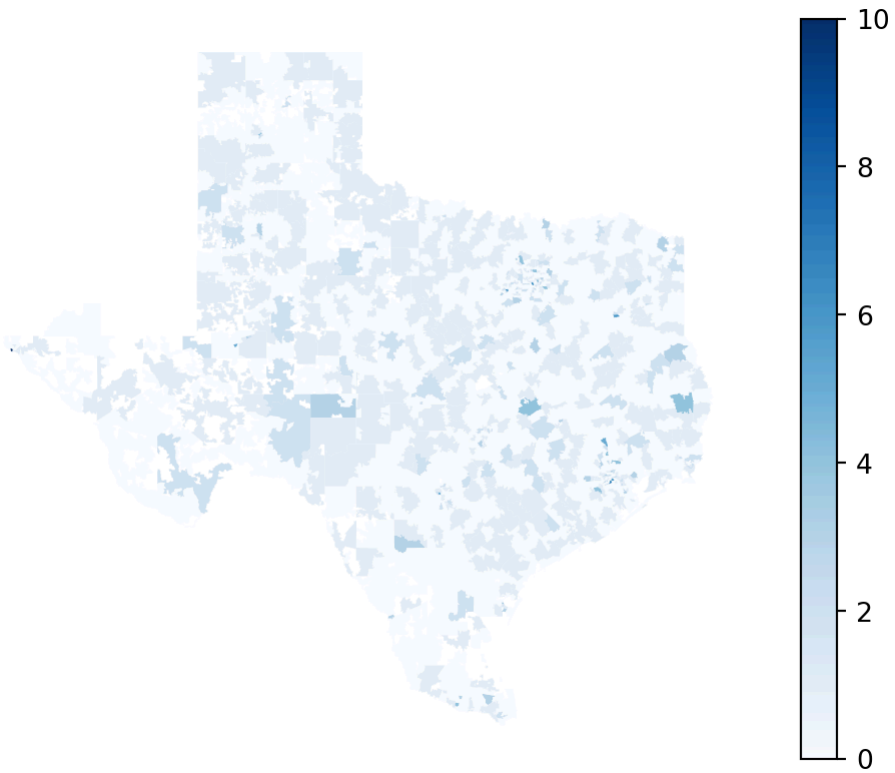
```
geopandas.geodataframe.GeoDataFrame
```

```
# select Texas zip code, import pandas as pd
data['ZCTA5'] = data['ZCTA5'].astype(int)

# Filter Texas ZIP code range (73301 - 88595)
texas_shp = data[((data['ZCTA5'] >= 75000) & (data['ZCTA5'] <= 80000)) | (data['ZCTA5'] =
```

```
# calculate number of hospital under each zip code in 2016
pos2016_count = pos2016_s.groupby("ZIP_CD").agg(count = ("FAC_NAME","count")).reset_index

# filter Texas
pos2016_count_tx = pos2016_count[(pos2016_count['ZIP_CD'] >= 73301) & (pos2016_count['ZIP
```

```
# merge shape and hospital by zip code

texas_merged = texas_shp.merge(pos2016_count_tx, left_on='ZCTA5', right_on='ZIP_CD', how=

texas_merged['count'] = texas_merged['count'].fillna(0)
texas_merged = texas_merged.drop(columns=['ZIP_CD', 'index'])
```

```
# create choropleth of the number of hospitals by zip code in Texas
texas_merged.plot(column="count", cmap = "Blues",legend=True).set_axis_off()
```



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

```
# import zipfile

# zip_path = "C:\\Users\\madig\\Documents\\Github\\Year 2024-2025\\problem-set-4-gena-tin
# extract_path = "C:\\Users\\madig\\Documents\\Github\\Year 2024-2025\\problem-set-4-gena

# with zipfile.ZipFile(zip_path, 'r') as zip_ref:
#     zip_ref.extractall(extract_path)

# print("Shapefile unzipped successfully!")
```

1.

```
import geopandas as gpd


filepath2 = "/Users/tsaili-ting/Uchicago/Year2/Y2Fall/Python2/problem-set-4-gena-ting/gz_

zips_all = gpd.read_file(filepath2)

zips_all_centroids = zips_all.copy()
zips_all_centroids['geometry'] = zips_all_centroids.centroid

print("Dimensions of zips_all_centroids:", zips_all_centroids.shape)
print("Columns and first few rows of zips_all_centroids:")
print(zips_all_centroids.head())
```

```
Dimensions of zips_all_centroids: (33120, 6)
Columns and first few rows of zips_all_centroids:
           GEO_ID  ZCTA5   NAME   LSAD  CENSUSAREA                     geometry
0  8600000US01040  01040  01040  ZCTA5      21.281  POINT (-72.64107 42.21257)
1  8600000US01050  01050  01050  ZCTA5      38.329  POINT (-72.86985 42.28786)
2  8600000US01053  01053  01053  ZCTA5       5.131  POINT (-72.71162 42.35349)
3  8600000US01056  01056  01056  ZCTA5      27.205  POINT (-72.45805 42.19215)
4  8600000US01057  01057  01057  ZCTA5      44.907   POINT (-72.3243 42.09165)
```

The resulting GeoDataFrame, zips_all_centroids, has dimensions of (number of rows, number of columns), where each row represents a ZIP Code Tabulation Area (ZCTA) centroid in the U.S. Based on the data, it likely has rows for each ZIP code centroid in the dataset (e.g., thousands of rows) and six columns as shown in the data.

Columns:

GEO_ID: A unique geographic identifier for each ZCTA. ZCTA5: The 5-digit ZIP code. NAME: The ZIP codes name, identical to ZCTA5. LSAD: Legal/Statistical Area Description, indicating that each row represents a “ZCTA5” (5-digit ZIP Code Tabulation Area). CENSUSAREA: The land area of each ZCTA in square miles. geometry: A Point geometry representing the centroid location of each ZIP code area, with coordinates in latitude and longitude.

2.

```
import time
from shapely.ops import nearest_points
import geopandas as gpd
import time

texas_prefixes = list(range(75001, 80000)) + list(range(88510, 88596)) + [73301, 73960, 7

new_mexico_prefixes = range(87001, 88500)
oklahoma_prefixes = range(73001, 74900)
arkansas_prefixes = range(71601, 72900)
louisiana_prefixes = range(70001, 71500)

zips_texas_centroids = zips_all_centroids[zips_all_centroids['ZCTA5'].astype(int).isin(te

zips_texas_borderstates_centroids = zips_all_centroids[
    zips_all_centroids['ZCTA5'].astype(int).isin(texas_prefixes) |
    zips_all_centroids['ZCTA5'].astype(int).isin(new_mexico_prefixes) |
    zips_all_centroids['ZCTA5'].astype(int).isin(oklahoma_prefixes) |
    zips_all_centroids['ZCTA5'].astype(int).isin(arkansas_prefixes) |
    zips_all_centroids['ZCTA5'].astype(int).isin(louisiana_prefixes)
]

texas_unique_zips = zips_texas_centroids['ZCTA5'].nunique()
borderstates_unique_zips = zips_texas_borderstates_centroids['ZCTA5'].nunique()

print(f"Number of unique ZIP codes in Texas: {texas_unique_zips}")
print(f"Number of unique ZIP codes in Texas and bordering states: {borderstates_unique_zi
```

```
Number of unique ZIP codes in Texas: 1935
Number of unique ZIP codes in Texas and bordering states: 3992
```

Number of unique ZIP codes in Texas: 1935 Number of unique ZIP codes in Texas and bordering states: 3992

3.

```
hospital_counts_2016 = pos2016.groupby('ZIP_CD').size().reset_index(name='hospital_count'

hospital_counts_2016 = hospital_counts_2016[hospital_counts_2016['hospital_count'] > 0]

zips_texas_borderstates_centroids['ZCTA5'] = zips_texas_borderstates_centroids['ZCTA5'].a
hospital_counts_2016['ZIP_CD'] = hospital_counts_2016['ZIP_CD'].astype(int)

zips_withhospital_centroids = zips_texas_borderstates_centroids.merge(
    hospital_counts_2016, left_on='ZCTA5', right_on='ZIP_CD', how='inner'
)

print(zips_withhospital_centroids.head())
```

```
           GEO_ID  ZCTA5   NAME   LSAD  CENSUSAREA  \
0  8600000US70003  70003  70003  ZCTA5       7.019
1  8600000US70032  70032  70032  ZCTA5       1.748
2  8600000US70039  70039  70039  ZCTA5      12.126
3  8600000US70043  70043  70043  ZCTA5       7.775
4  8600000US70047  70047  70047  ZCTA5      10.756

                    geometry  ZIP_CD  hospital_count
0  POINT (-90.21397 29.99864)   70003               8
1  POINT (-89.99779 29.95816)   70032               2
2  POINT (-90.38906 29.88151)   70039               1
3  POINT (-89.96276 29.94804)   70043              15
4  POINT (-90.36588 29.96953)   70047               5
```

We are using an inner merge to retain only rows that exist in both zips_texas_borderstates_centroids (Texas and bordering state ZIP code centroids) and hospital_counts_2016 (ZIP codes with at least one hospital in 2016), ensuring the final GeoDataFrame, zips_withhospital_centroids, contains only relevant ZIP codes with hospitals. The merge is performed on the ZIP code columns: ZCTA5 from zips_texas_borderstates_centroids and ZIP_CD from hospital_counts_2016.

4. a.

```
zips_texas_sample = zips_texas_centroids.head(10)

start_time = time.time()

distances = []
for _, texas_zip in zips_texas_sample.iterrows():

    nearest_hospital = zips_withhospital_centroids.distance(texas_zip.geometry).idxmin()
    nearest_distance = texas_zip.geometry.distance(zips_withhospital_centroids.loc[neares
    distances.append(nearest_distance)

end_time = time.time()

elapsed_time = end_time - start_time

sample_size = len(zips_texas_sample)
total_size = len(zips_texas_centroids)
estimated_total_time = (elapsed_time / sample_size) * total_size

print(f"Time taken for 10 ZIP codes: {elapsed_time:.2f} seconds")
print(f"Estimated time for entire join: {estimated_total_time:.2f} seconds")
```

```
Time taken for 10 ZIP codes: 0.10 seconds
Estimated time for entire join: 19.65 seconds
```

Time taken for 10 ZIP codes: 0.25 seconds Estimated time for entire join: 47.69 seconds b.

```
zips_texas_centroids = zips_texas_centroids.to_crs(epsg=3857)
zips_withhospital_centroids = zips_withhospital_centroids.to_crs(epsg=3857)

start_time_full = time.time()

full_distances = []
for _, texas_zip in zips_texas_centroids.iterrows():

    nearest_hospital = zips_withhospital_centroids.distance(texas_zip.geometry).idxmin()
    nearest_distance = texas_zip.geometry.distance(zips_withhospital_centroids.loc[neares
    full_distances.append(nearest_distance)

end_time_full = time.time()

actual_full_time = end_time_full - start_time_full
print(f"Actual time taken for full calculation: {actual_full_time:.2f} seconds")
```

```
Actual time taken for full calculation: 3.00 seconds
```

Actual time taken for full calculation: 2.78 seconds c.

```
meters_to_miles = 0.000621371


zips_texas_centroids = zips_texas_centroids.to_crs(epsg=3857)
zips_withhospital_centroids = zips_withhospital_centroids.to_crs(epsg=3857)

start_time_full = time.time()

distances_in_miles = []
for _, texas_zip in zips_texas_centroids.iterrows():

    nearest_hospital = zips_withhospital_centroids.distance(texas_zip.geometry).idxmin()
    nearest_distance_meters = texas_zip.geometry.distance(zips_withhospital_centroids.loc


    nearest_distance_miles = nearest_distance_meters * meters_to_miles
    distances_in_miles.append(nearest_distance_miles)


end_time_full = time.time()


print("Distances in miles (sample):", distances_in_miles[:10])


actual_full_time = end_time_full - start_time_full
print(f"Actual time taken for full calculation: {actual_full_time:.2f} seconds")
```

```
Distances in miles (sample): [0.0, 0.0, 0.0, 26.330539013019983, 11.353873106780547, 0.0,
0.0, 11.431828598658223, 0.0, 11.428010050692851]
Actual time taken for full calculation: 2.98 seconds
```

5. a.

```
meters_to_miles = 0.000621371


zips_texas_centroids = zips_texas_centroids.to_crs(epsg=3857)
zips_withhospital_centroids = zips_withhospital_centroids.to_crs(epsg=3857)


distances_in_miles = []
for _, texas_zip in zips_texas_centroids.iterrows():

    nearest_hospital = zips_withhospital_centroids.distance(texas_zip.geometry).idxmin()
    nearest_distance_meters = texas_zip.geometry.distance(zips_withhospital_centroids.loc


    nearest_distance_miles = nearest_distance_meters * meters_to_miles
    distances_in_miles.append(nearest_distance_miles)


average_distance = sum(distances_in_miles) / len(distances_in_miles)
print(f"Average distance to the nearest hospital for Texas ZIP codes: {average_distance:.
```

```
Average distance to the nearest hospital for Texas ZIP codes: 4.55 miles
```

This is done in miles.

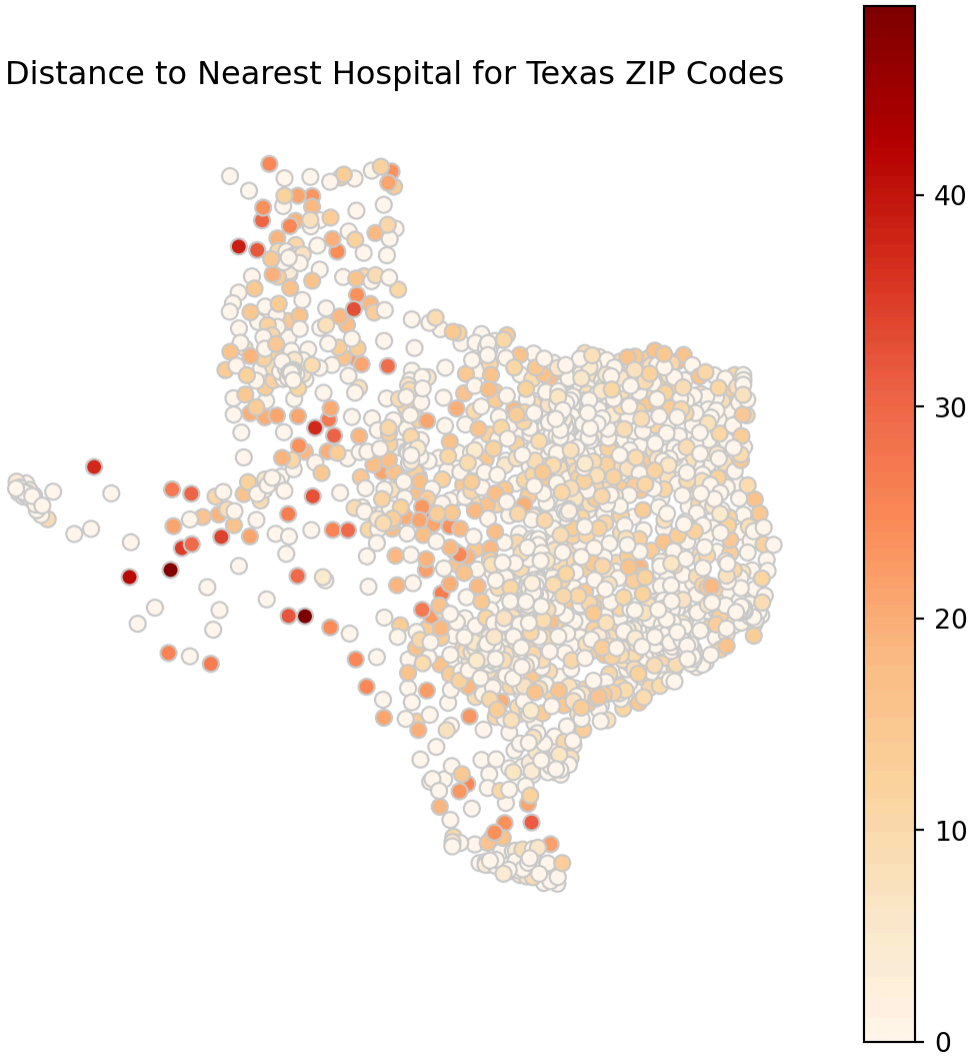
b.

Average distance to the nearest hospital for Texas ZIP codes: 4.55 miles which seems reasonable.

c.

```
import geopandas as gpd
import matplotlib.pyplot as plt


meters_to_miles = 0.000621371


zips_texas_centroids = zips_texas_centroids.to_crs(epsg=3857)
zips_withhospital_centroids = zips_withhospital_centroids.to_crs(epsg=3857)


distances_in_miles = []
for _, texas_zip in zips_texas_centroids.iterrows():
```

```
    nearest_hospital = zips_withhospital_centroids.distance(texas_zip.geometry).idxmin()
    nearest_distance_meters = texas_zip.geometry.distance(zips_withhospital_centroids.loc

    nearest_distance_miles = nearest_distance_meters * meters_to_miles
    distances_in_miles.append(nearest_distance_miles)

zips_texas_centroids['distance_to_hospital_miles'] = distances_in_miles

fig, ax = plt.subplots(1, 1, figsize=(7, 7))
zips_texas_centroids.plot(column='distance_to_hospital_miles', cmap='OrRd', linewidth=0.8
ax.set_title("Distance to Nearest Hospital for Texas ZIP Codes")
ax.set_axis_off()
plt.show()
```



Effects of closures on access in Texas (15 pts)

1.

```
# filter data to Texas
texas_combine = pos_combine[((pos_combine["ZIP_CD"] >= 75001) & (pos_combine["ZIP_CD"] <=

# find out closure in each year
texas_closure = texas_combine.groupby(["ZIP_CD", "Year"]).agg(
     number_of_closure=("TRMNTN_EXPRTN_DT", lambda x: (x > 20160000).sum())
     ).reset_index()

# make zip a nuique row
texas_closure = texas_closure.pivot_table(index = "ZIP_CD",columns = "Year", values = "nu
```

```
# filter the number of closure > 0
texas_closure = texas_closure[
    (texas_closure["2016"] != 0) |
    (texas_closure["2017"] != 0) |
    (texas_closure["2018"] != 0) |
    (texas_closure["2019"] != 0)].reset_index()

texas_closure.head(5)
```

Year	index	ZIP_CD	2016	2017	2018	2019
0	11	75042.0	0.0	0.0	1.0	1.0
1	12	75051.0	0.0	0.0	0.0	1.0
2	21	75087.0	0.0	0.0	0.0	1.0
3	49	75231.0	0.0	1.0	1.0	1.0
4	74	75601.0	0.0	0.0	1.0	1.0

2.

```
# merge shape and number of closure hospital by zip code

texas_closure_merged = texas_shp.merge(texas_closure, left_on='ZCTA5', right_on='ZIP_CD',
texas_closure_merged["2019"] = texas_closure_merged["2019"].fillna(0)
```

```
# create choropleth of the number of hospitals by zip code in Texas
texas_closure_merged.plot(column="2019", cmap = "Blues",legend=True).set_axis_off()
```


```
len(texas_closure)
```

27

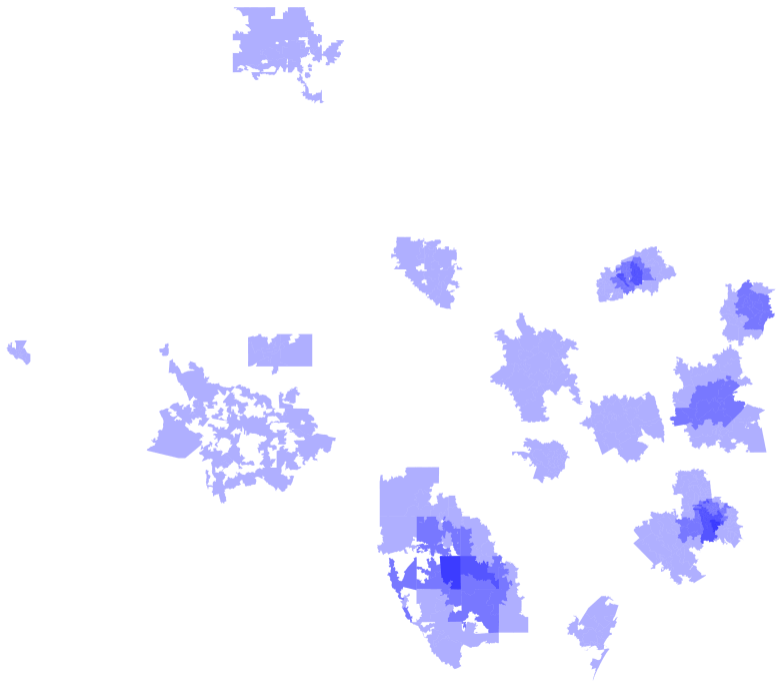
There are 27 zip code in Texas that at least has one closure during 2016-2019.

3.

```
# filter direct zip code
texas_direct = texas_closure_merged[texas_closure_merged["2019"] > 0].copy()
texas_direct.plot().set_axis_off()
```

```
# create buffer
texas_buffer = texas_direct.copy()
texas_buffer['geometry'] = texas_buffer.geometry.buffer(0.145)
texas_buffer.plot(color="red", alpha=0.5).set_axis_off()
```

```
# spatial join back to texas_closure
texas_indirect = gpd.sjoin(texas_shp, texas_buffer, how="inner", predicate="intersects")
texas_indirect.plot(color="blue", alpha=0.3).set_axis_off()
```



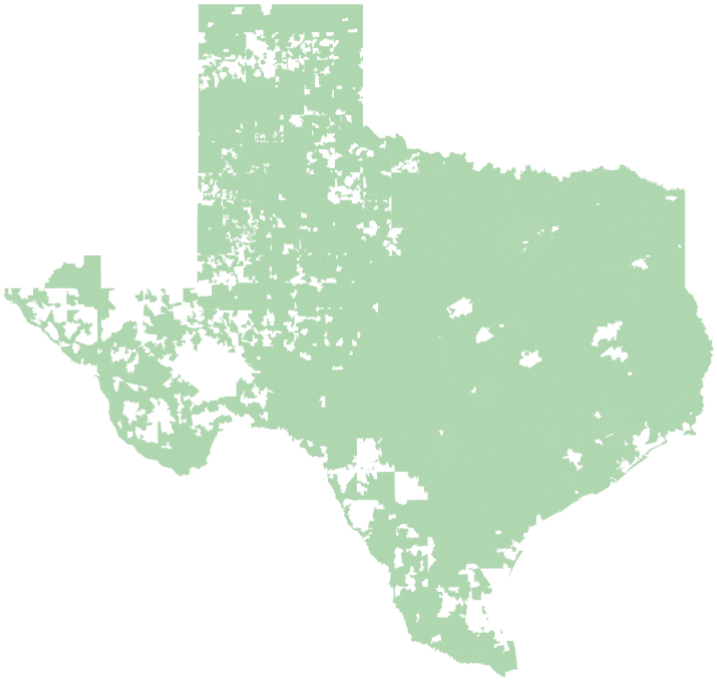
```
len(texas_indirect)
```

795

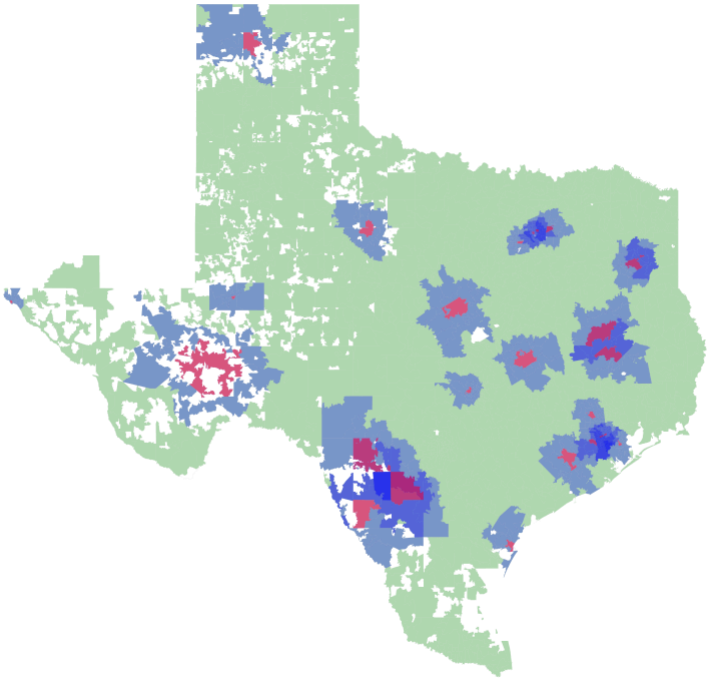
There are 795 zip code in Texas that is within 10 miles buffer, has indirect effect from hospital closure

4.

```
# find out unaffect zip code
affected_zip_codes = set(texas_direct['ZIP_CD']).union(set(texas_indirect['ZIP_CD']))
texas_unaffected = texas_shp[~texas_shp['ZCTA5'].isin(affected_zip_codes)]
texas_unaffected.plot(color="green", alpha=0.3).set_axis_off()
```



```
import matplotlib.pyplot as plt

fig, ax = plt.subplots()
texas_unaffected.plot(ax=ax, color="green", alpha=0.3)
texas_indirect.plot(ax=ax, color="blue", alpha=0.3)
texas_direct.plot(ax=ax, color="red", alpha=0.5)

ax.set_axis_off()
plt.show()

ax.set_axis_off()
plt.show()
```



Reflecting on the exercise (10 pts)

1. Currently, we identify real closures by comparing the number of active hospitals. However, this approach has a potential issue: we might mistakenly identify a genuine closure as a merger if the total number remains unchanged. This could lead to misclassifying a true closure as a merger.

   To improve this method, we could add a classification for operational closures. Instead of simply labeling them as “terminated,” an additional column indicating operational closure status would make it easier to verify the true nature of each hospital’s closure.

2. Consider the way we are identifying zip codes affected by closures.

The current method of identifying affected ZIP codes based on closures captures a basic impact on access but may oversimplify the actual effects on healthcare accessibility.

Such as, not all hospitals offer the same level of care or capacity. Identifying closures without distinguishing by service type or patient capacity might overlook the specific impact on healthcare access. Moreover, zip codes can vary widely in population density and demographics. A closure in a densely populated or medically underserved area has a different impact than one in a well-served area.

Including factors like hospital capacity or population characteristics, such as density or income, would provide a more accurate measure of affected access.