Pset 5

Shuping Wu, Sijie Wu

2024-11-09

Due 11/9 at 5:00PM Central. Worth 100 points + 10 points extra credit.

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 : Shuping Wu, shupingw
    - Partner 2 : Sijie Wu, sijiewu
3. Partner 1 will accept the `ps5` 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 `ps5.qmd` to an PDF file to make `ps5.pdf`,
    - The PDF should not be more than 25 pages. Use `head()` and re-size figures when appropriate.
8. (Partner 1): push `ps5.qmd` and `ps5.pdf` to your github repo.
9. (Partner 1): submit `ps5.pdf` via Gradescope. Add your partner on Gradescope.
10. (Partner 1): tag your submission in Gradescope

```
import pandas as pd
import altair as alt
import time

import warnings
warnings.filterwarnings('ignore')
alt.renderers.enable("png")
```

```
RendererRegistry.enable('png')
```

```
import requests
from bs4 import BeautifulSoup
from bs4.element import Tag
```

Step 1: Develop initial scraper and crawler

1. Scraping (PARTNER 1)

```
# Step 1: Fetch the webpage
url = "https://oig.hhs.gov/fraud/enforcement/"
response = requests.get(url)
soup = BeautifulSoup(response.text, 'html.parser')

# Step 2: Scrape required data
titles = []
dates = []
categories = []
links = []

for action in soup.select('header.usa-card__header'):

    title_tag = action.select_one('h2.usa-card__heading a')
    title = title_tag.text.strip()
    link = "https://oig.hhs.gov" + title_tag['href']
    date_tag = action.select_one('div.font-body-sm.margin-top-1 span')
    date = date_tag.text.strip()
    cat_tag = action.select_one('div.font-body-sm.margin-top-1 ul')
    categories.append(cat_tag.text.strip())
    titles.append(title)
```

```
    links.append(link)
    dates.append(date)
```

```
df = pd.DataFrame({
    'Title': titles,
    'Date': dates,
    'Category': categories,
    'Link': links
})
print(df.head())
```

```
                                               Title              Date  \
0  Pharmacist and Brother Convicted of $15M Medic...  November 8, 2024
1  Boise Nurse Practitioner Sentenced To 48 Month...  November 7, 2024
2  Former Traveling Nurse Pleads Guilty To Tamper...  November 7, 2024
3  Former Arlington Resident Sentenced To Prison ...  November 7, 2024
4  Paroled Felon Sentenced To Six Years For Fraud...  November 7, 2024

                    Category  \
0  Criminal and Civil Actions
1  Criminal and Civil Actions
2  Criminal and Civil Actions
3  Criminal and Civil Actions
4  Criminal and Civil Actions

                                                Link
0  https://oig.hhs.gov/fraud/enforcement/pharmaci...
1  https://oig.hhs.gov/fraud/enforcement/boise-nu...
2  https://oig.hhs.gov/fraud/enforcement/former-t...
3  https://oig.hhs.gov/fraud/enforcement/former-a...
4  https://oig.hhs.gov/fraud/enforcement/paroled-...
```

2. Crawling (PARTNER 1)

```
agencies = []
for link in df['Link']:
    detail_response = requests.get(link)
    detail_soup = BeautifulSoup(detail_response.text, 'html.parser')
    agency_tag = detail_soup.select(
```

```
        'ul.usa-list.usa-list--unstyled.margin-y-2 li')
    second_li = agency_tag[1]
    if not isinstance(second_li.contents[1], Tag):
        agency = second_li.contents[1].strip()
    else:
        agency = 'NaN'
    agencies.append(agency)
# df['Agency'] = agencies
print(agencies)
```

```
['U.S. Department of Justice', "November 7, 2024; U.S. Attorney's Office,
District of Idaho", "U.S. Attorney's Office, District of Massachusetts",
"U.S. Attorney's Office, Eastern District of Virginia", "U.S. Attorney's
Office, Middle District of Florida", "U.S. Attorney's Office, Western
District of Texas", "U.S. Attorney's Office, Eastern District of Michigan",
"U.S. Attorney's Office, Eastern District of Tennessee", "U.S. Attorney's
Office, Northern District of Texas", 'U.S. Department of Justice', 'State of
South Carolina', "U.S. Attorney's Office, Eastern District of Missouri",
"U.S. Attorney's Office, Middle District of Tennessee", 'U.S. Department of
Justice', 'State of New Mexico', 'State of Tennessee', 'NaN', 'Ohio', 'NaN',
'State of Massachusetts']
```

```
df['Agency'] = agencies
print(df.head())
```

```
                                               Title              Date  \
0  Pharmacist and Brother Convicted of $15M Medic...  November 8, 2024
1  Boise Nurse Practitioner Sentenced To 48 Month...  November 7, 2024
2  Former Traveling Nurse Pleads Guilty To Tamper...  November 7, 2024
3  Former Arlington Resident Sentenced To Prison ...  November 7, 2024
4  Paroled Felon Sentenced To Six Years For Fraud...  November 7, 2024

                    Category  \
0  Criminal and Civil Actions
1  Criminal and Civil Actions
2  Criminal and Civil Actions
3  Criminal and Civil Actions
4  Criminal and Civil Actions

                                                Link  \
0  https://oig.hhs.gov/fraud/enforcement/pharmaci...
1  https://oig.hhs.gov/fraud/enforcement/boise-nu...
```

```
2  https://oig.hhs.gov/fraud/enforcement/former-t...
3  https://oig.hhs.gov/fraud/enforcement/former-a...
4  https://oig.hhs.gov/fraud/enforcement/paroled-...

                                                Agency
0                          U.S. Department of Justice
1  November 7, 2024; U.S. Attorney's Office, Dist...
2  U.S. Attorney's Office, District of Massachusetts
3  U.S. Attorney's Office, Eastern District of Vi...
4  U.S. Attorney's Office, Middle District of Flo...
```

Step 2: Making the scraper dynamic

1. Turning the scraper into a function

- a. Pseudo-Code (PARTNER 2) First, check date input.

Then, create a loop to read in the current url.

Create a for loop to loop in the current page, checking the date of each entry every time, stop when reached date limit.

Proceed to the next page if not reached start date.

- b. Create Dynamic Scraper (PARTNER 2)

```
from datetime import datetime
import time


def scrape_enforcement_actions(month, year):
    if year < 2013:
        print("Year must be 2013 or later.")
        return

    start_date = datetime(year, month, 1)

    base_url = "https://oig.hhs.gov/fraud/enforcement/?page="
    actions_data = []
    page_number = 1
    done = False

    while not done:
```

```
        url = f"{base_url}{page_number}"
        response = requests.get(url)
        soup = BeautifulSoup(response.text, 'html.parser')

        actions = soup.select('header.usa-card__header')

        for action in actions:
            date_tag = action.select_one('div.font-body-sm.margin-top-1
↪  span')
            date = date_tag.text.strip()
            print(date)
            action_date = datetime.strptime(date, "%B %d, %Y")

            if start_date <= action_date:
                title_tag = action.select_one('h2.usa-card__heading a')
                title = title_tag.text.strip()
                print(title)
                link = "https://oig.hhs.gov" + title_tag['href']
                category = action.select_one(

                    ↪  'li.display-inline-block.usa-tag.text-no-lowercase').text.strip()
                agency = scrape_agency(link)
                print(agency)

                actions_data.append({
                    'Title': title,
                    'Date': action_date.strftime("%Y-%m-%d"),
                    'Category': category,
                    'Link': link,
                    'Agency': agency
                })
            elif action_date < start_date:
                done = True
                break

        page_number += 1
        time.sleep(1)

        # Save to CSV
    df = pd.DataFrame(actions_data)
    file_name = f"enforcement_actions_{year}_{month}.csv"
    df.to_csv(file_name, index=False)
```

```
    print(f"Data saved to {file_name}")
```

```
def scrape_agency(link):
    detail_response = requests.get(link)
    detail_soup = BeautifulSoup(detail_response.text, 'html.parser')
    agency_tag = detail_soup.select(
        'ul.usa-list.usa-list--unstyled.margin-y-2 li')
    second_li = agency_tag[1]
    if not isinstance(second_li.contents[1], Tag):
        agency = second_li.contents[1].strip()
    else:
        agency = 'NaN'
    return agency
```

- c. Test Partner's Code (PARTNER 1)

```
#scrape_enforcement_actions(1, 2021)
```

Step 3: Plot data based on scraped data

1. Plot the number of enforcement actions over time (PARTNER 2)

```
filepath = "enforcement_actions_2021_1.csv"
data = pd.read_csv(filepath)
data.head()
```

	Title	Date	Category	Link
0	Former Arlington Resident Sentenced To Prison ...	2024-11-07	Criminal and Civil Actions	https:/
1	Paroled Felon Sentenced To Six Years For Fraud...	2024-11-07	Criminal and Civil Actions	https:/
2	Former Licensed Counselor Sentenced For Defrau...	2024-11-06	Criminal and Civil Actions	https:/
3	Macomb County Doctor And Pharmacist Agree To P...	2024-11-04	Criminal and Civil Actions	https:/
4	Rocky Hill Pharmacy And Its Owners Indicted Fo...	2024-11-04	Criminal and Civil Actions	https:/

```
import altair as alt
from altair_saver import save
```

```
# Change "Date" into datetime format
data['Date'] = pd.to_datetime(data['Date'])

# Aggregate by month and year
monthly_actions = data.groupby(data['Date'].dt.to_period(
    "M")).size().reset_index(name='count')
monthly_actions['Date'] = monthly_actions['Date'].dt.to_timestamp()

# Plot with Altair
chart1 = alt.Chart(monthly_actions).mark_line().encode(
    x=alt.X('Date:T', title='Date', axis=alt.Axis(format='%Y-%m')),
    y=alt.Y('count:Q', title='Number of Enforcement Actions'),
    tooltip=['Date:T', 'count:Q']
).properties(
    title='Number of Enforcement Actions Over Time (Monthly Aggregation)',
    width=400,
    height=200
)
chart1.save(

    ↪ '/Users/wsjsmac/Desktop/Autumn/PPHA_30538/mine/Pset_5/pair-from-git/problemset5/chart
```

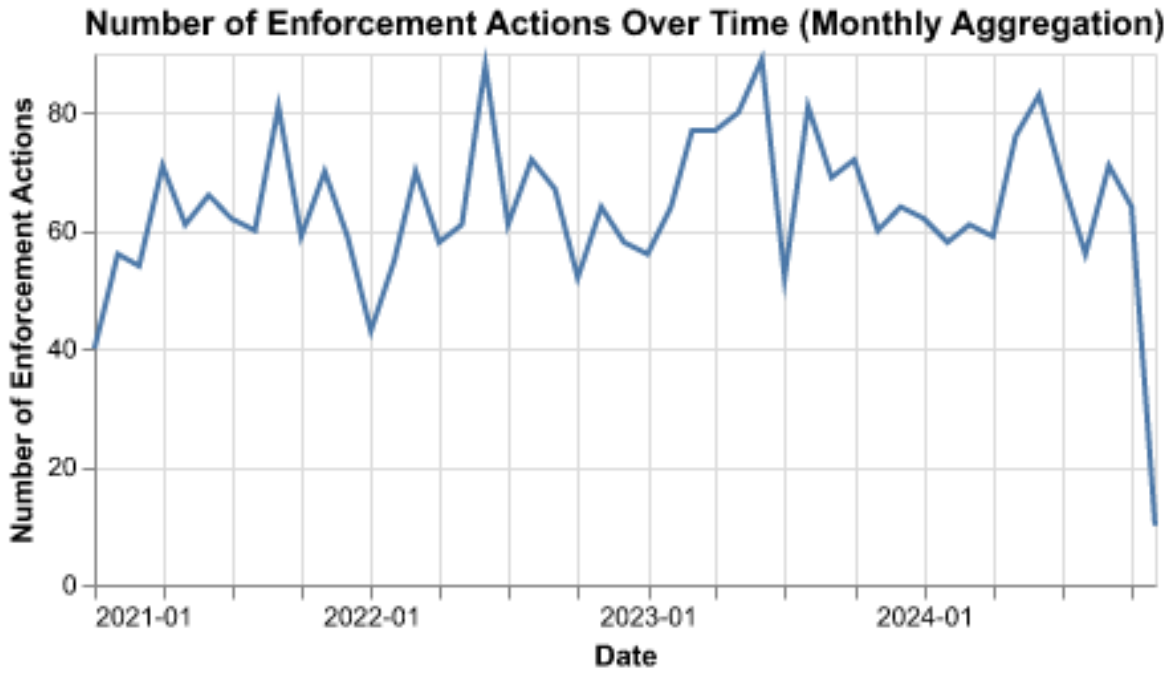


Figure 1: Number of Enforcement Actions Over Time

2. Plot the number of enforcement actions categorized: (PARTNER 1)

- based on "Criminal and Civil Actions" vs. "State Enforcement Agencies"

```
# Change "Date" into datetime format
data['Date'] = pd.to_datetime(data['Date'])

# Aggregate by month and year, and by category
monthly_actions = (
    data[data['Category'].isin(
        ['Criminal and Civil Actions', 'State Enforcement Agencies'])]
    .groupby([data['Date'].dt.to_period("M"), 'Category'])
    .size()
    .reset_index(name='Count')
)

# Convert 'Date' back to timestamp for plotting
monthly_actions['Date'] = monthly_actions['Date'].dt.to_timestamp()

# Plot with Altair
```

```
chart2 = alt.Chart(monthly_actions).mark_line().encode(
    x=alt.X('Date:T', title='Date', axis=alt.Axis(format='%Y-%m')),
    y=alt.Y('Count:Q', title='Number of Enforcement Actions'),
    color=alt.Color('Category:N', title='Category'),
    tooltip=['Date:T', 'Count:Q', 'Category:N']
).properties(
    title='Number of Enforcement Actions Over Time by Category',
    width=400,
    height=200
).interactive()

# chart2

chart2.save(

    ↪ '/Users/wsjsmac/Desktop/Autumn/PPHA_30538/mine/Pset_5/pair-from-git/problemset5/chart
```

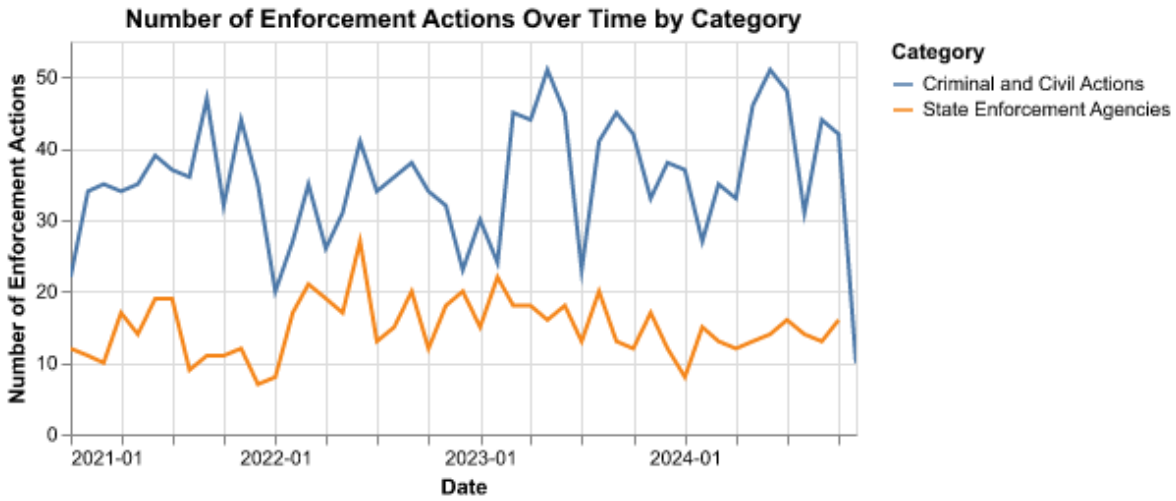


Figure 2: Number of Enforcement Actions Over Time by Category

- based on five topics

```
# Change "Date" into datetime format
data['Date'] = pd.to_datetime(data['Date'])

data_criminal_civil = data[data['Category'].isin(
    ['Criminal and Civil Actions'])]
```

```
# Define a function to categorize based on keywords in 'Title'


def categorize_title(title):
    if any(keyword in title.lower() for keyword in ['drug', 'narcotics',
     ↪  'enforcement', 'trafficking']):
        return 'Drug Enforcement'
    elif any(keyword in title.lower() for keyword in ['health', 'insurance',
     ↪  'hospital', 'medicare', 'medicaid']):
        return 'Health Care Fraud'
    elif any(keyword in title.lower() for keyword in ['bank', 'financial',
     ↪  'investment', 'securities']):
        return 'Financial Fraud'
    elif any(keyword in title.lower() for keyword in ['bribery',
     ↪  'corruption', 'bribe']):
        return 'Bribery/Corruption'
    else:
        return 'Other'


# Apply categorization function to the Title column
data_criminal_civil['Topic'] = data_criminal_civil['Title'].apply(
    categorize_title)

data_criminal_civil.head()
```

	Title	Date	Category	Link
0	Former Arlington Resident Sentenced To Prison ...	2024-11-07	Criminal and Civil Actions	https:/
1	Paroled Felon Sentenced To Six Years For Fraud...	2024-11-07	Criminal and Civil Actions	https:/
2	Former Licensed Counselor Sentenced For Defrau...	2024-11-06	Criminal and Civil Actions	https:/
3	Macomb County Doctor And Pharmacist Agree To P...	2024-11-04	Criminal and Civil Actions	https:/
4	Rocky Hill Pharmacy And Its Owners Indicted Fo...	2024-11-04	Criminal and Civil Actions	https:/

```
# Aggregate by month and year, and by topic
monthly_actions = (
    data_criminal_civil
    .groupby([data_criminal_civil['Topic'],
 ↪  data_criminal_civil['Date'].dt.to_period("M")])
    .size()
```

```
    .reset_index(name='Count')
)

# Convert 'Date' back to timestamp for plotting
monthly_actions['Date'] = monthly_actions['Date'].dt.to_timestamp()

# Plot with Altair
chart3 = alt.Chart(monthly_actions).mark_line().encode(
    x=alt.X('Date:T', title='Date', axis=alt.Axis(format='%Y-%m')),
    y=alt.Y('Count:Q', title='Number of Categories'),
    color=alt.Color('Topic:N', title='Topic'),
    tooltip=['Date:T', 'Count:Q', 'Topic:N']
).properties(
    title='Number of Criminal and Civil Actions Over Time by Topic',
    width=400,
    height=200
).interactive()

# chart3

chart3.save(

    ↪ '/Users/wsjsmac/Desktop/Autumn/PPHA_30538/mine/Pset_5/pair-from-git/problemset5/chart
```

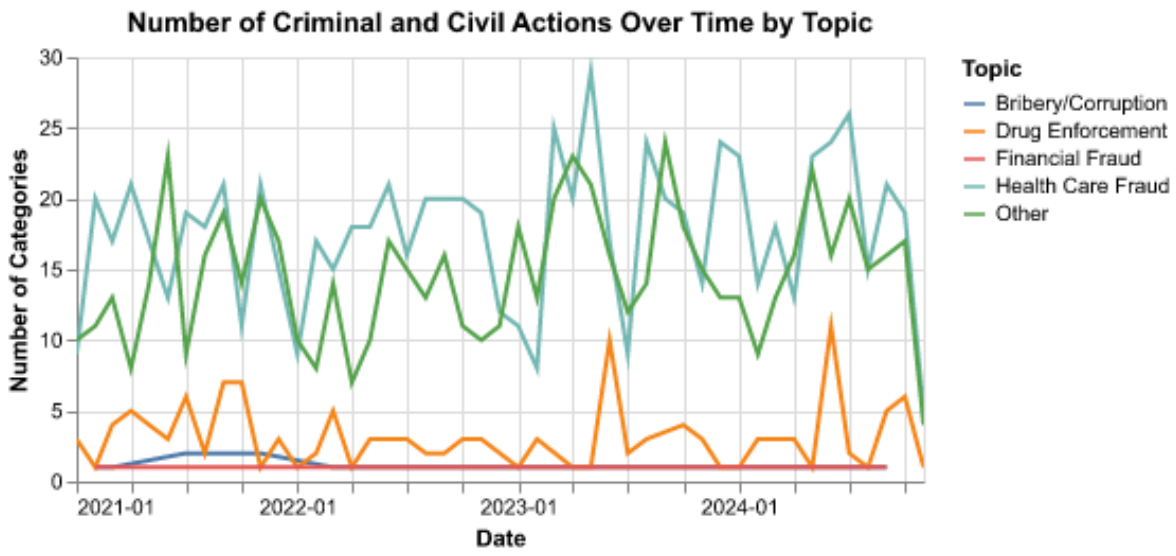


Figure 3: Number of Criminal and Civil Actions Over Time by Topic

Step 4: Create maps of enforcement activity

1. Map by State (PARTNER 1)

```
filepath = "enforcement_actions_2021_1.csv"
data = pd.read_csv(filepath)
```

```
data_state = data[data["Agency"].str.contains("State of", case=False,
 ↪  na=False)]
data_state = data_state.reset_index(drop=True)
```

```
import geopandas as gpd
# Load the shapefile
# Replace with the actual path to your shapefile
states_gdf = gpd.read_file("cb_2018_us_state_500k/cb_2018_us_state_500k.shp")
#states_gdf.head()
```

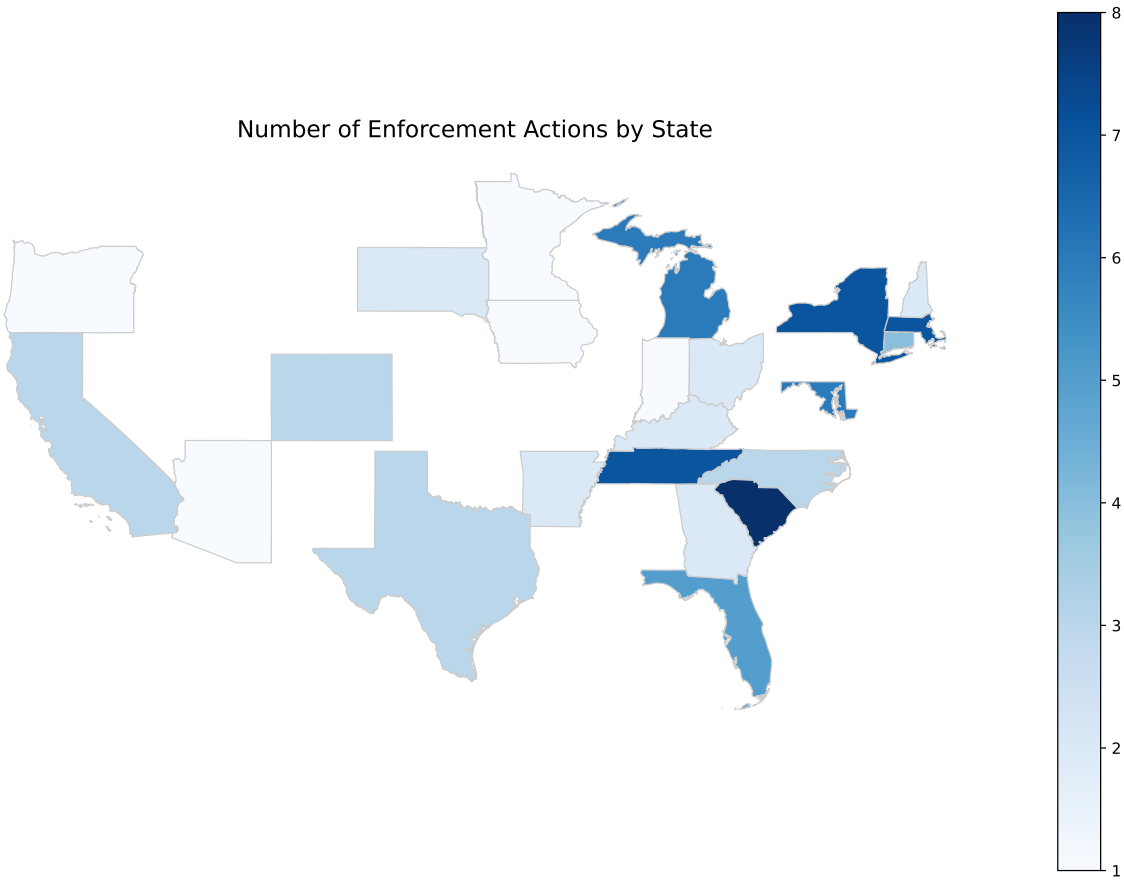
```
state_count = (
    data_state
    .groupby('Agency')
    .size()
    .reset_index(name='Count')
)
#state_count.head()
```

```
state_count["Agency_Name"] = state_count['Agency'].str.replace(
    r'^State of ', '', regex=True)
```

```
# Ensure state names/abbreviations match in both DataFrames
merged_gdf = states_gdf.merge(
    state_count, left_on='NAME', right_on='Agency_Name', how='right')

#merged_gdf.head()
```

```
import matplotlib.pyplot as plt

# Plotting the map
fig, ax = plt.subplots(1, 1, figsize=(15, 10))
```

```
merged_gdf.plot(column='Count', cmap='Blues', linewidth=0.8,
                edgecolor='0.8', legend=True, ax=ax)
ax.set_title("Number of Enforcement Actions by State", fontsize=15)
ax.set_axis_off()
plt.show()
```



2. Map by District (PARTNER 2)

```
import geopandas as gpd
filepath1 =
 ↪  "/Users/wsjsmac/Desktop/Autumn/PPHA_30538/mine/Pset_5/pair-from-git/problemset5/enforceme
data = pd.read_csv(filepath1)
```

```
filepath =
 ↪  "/Users/wsjsmac/Desktop/Autumn/PPHA_30538/mine/Pset_5/pair-from-git/problemset5/US
 ↪  Attorney Districts Shapefile
 ↪  simplified_20241109/geo_export_c632cc9b-772f-489b-8052-e9829f5262fb.shp"
district = gpd.read_file(filepath)
```

```
#district.head()
```

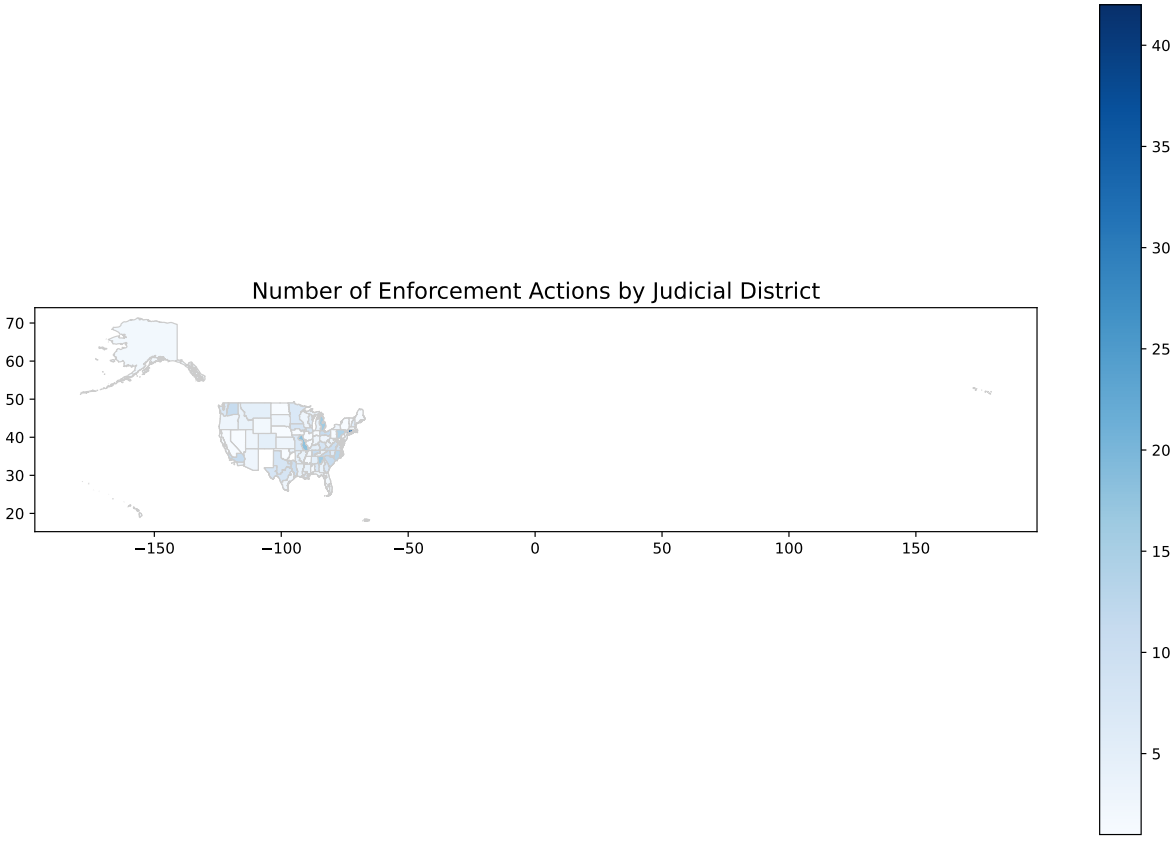
```
data_district = data[data["Agency"].str.contains(
    "District", case=False, na=False)]
data_district = data_district.reset_index(drop=True)

district_count = (
    data_district
    .groupby('Agency')
    .size()
    .reset_index(name='Count')
)
```

```
def extract_district(agency):
    if 'Attorney’s Office, ' in agency:  # Case with 'Attorney’s Office, '
        return agency.split("Attorney’s Office, ")[-1]
    elif 'Attorney\'s Office, ' in agency:  # Case with 'Attorney\'s Office,
     ↪  '
        return agency.split("Attorney's Office, ")[-1]
    elif 'U.S. Attorney’s Office;' in agency:  # Case with 'U.S. Attorney’s
     ↪  Office;'
        return agency.split("U.S. Attorney’s Office;")[-1]
    elif 'U.S. Attorney’s Office' in agency:  # Case with 'U.S. Attorney’s
     ↪  Office;'
        return agency.split("U.S. Attorney’s Office")[-1]
    elif "U.S. Attorney General," in agency:
        return agency.split("U.S. Attorney General,")[-1]
    elif "U.S. AttorneyÍs Office," in agency:
        return agency.split("U.S. AttorneyÍs Office,")[-1]
    else:  # For cases without the expected prefixes
        return agency


# Apply the function to the 'Agency' column
```

```
district_count['Agency_Name'] = district_count['Agency'].apply(
    extract_district)

#district_count.head()
```

```
# Ensure state names/abbreviations match in both DataFrames
merged = district.merge(district_count, left_on='judicial_d',
                        right_on='Agency_Name', how='right')
#merged.head()
```

```
import matplotlib.pyplot as plt
# Plotting the map
# Set limits to focus on the contiguous U.S.
ax.set_xlim(-130, -60)  # Adjust these limits based on the actual extent
ax.set_ylim(20, 55)
fig, ax = plt.subplots(1, 1, figsize=(15, 10), dpi=150)
merged.plot(column='Count', cmap='Blues', linewidth=0.8, edgecolor='0.8',
 ↪  legend=True, ax=ax, aspect=1.5)
ax.set_title("Number of Enforcement Actions by Judicial District",
 ↪  fontsize=15)
plt.show()
```



Extra Credit

1. Merge zip code shapefile with population

```
#
```

2. Conduct spatial join

3. Map the action ratio in each district

```
#
```