PS5 Andy Fan Will Sigal

PS4: Due Sat Nov 9 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 (name and cnet ID): Andy Fan, fanx
    - Partner 2 (name and cnet ID): Will Sigal, Wsigal
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:

AF WS

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: (Andy:0); (Will:0) Late coins left after submission: (Andy: 3) ; (Will: 4)
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

```
### SETUP
import pandas as pd
import altair as alt
import time
import os
import warnings
import geopandas as gpd
import numpy as np
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
import requests
from bs4 import BeautifulSoup
```

(30 points) Step 1: Develop initial scraper and crawler

1. (Partner 1) Scraping: Go to the first page of the HHS OIG's "Enforcement Actions"page and scrape and collect the following into a dataset: ▪ Title of the enforcement action ▪ Date ▪ Category (e.g, "Criminal and Civil Actions") ▪ Link associated with the enforcement action Collect your output into a tidy dataframe and print its head.

```
### making soup
url1 = 'https://oig.hhs.gov/fraud/enforcement'
response1 = requests.get(url1)
soup1 = BeautifulSoup(response1.content, 'lxml')
```

```
### find title of enforcements
li_blocks = soup1.find_all('h2') # h2 classes with nested 'a' titles.
 ↪  li_blocks[2:21] are the 20 datapoints
li_titles = []
for h2 in li_blocks:
    for a_tag in h2.find_all('a'):
        li_titles.append(a_tag)
li_titles[0:5]
df_title = pd.DataFrame(li_titles) # dataframe with titles
df_title.columns = ['title']
```

Title: each title is a 'a' class, under h2 class. 'h2 class' under 'header class' under 'div class' under 'li class'

```
### find date
span_blocks = soup1.find_all('span', attrs={'class': 'text-base-dark
 ↪  padding-right-105'})
span_blocks[0:5]
df_date = pd.DataFrame(span_blocks)
df_date.columns = ['date']
#asked chat gpt 'how to i search for 'span' class with attribute xxx'
```

Date: is under 'span' class

```
### find category
li_blocks_dt = soup1.find_all('li', attrs={'class': 'display-inline-block
 ↪  usa-tag text-no-lowercase text-base-darkest bg-base-lightest
 ↪  margin-right-1'})
li_blocks[0:5]
df_category = pd.DataFrame(li_blocks_dt)
df_category.columns = ['category']
```

Category: each category is a li class (search by attribute).'l1 class' under 'ul class' under 'div class' under 'header class'

```
### find link
#use the list of titles from p1 and extract href
link_blocks = [link.get('href') for link in li_titles]
df_link = pd.DataFrame(link_blocks)
df_link.columns = ['link']
df_link['link'] = "https://oig.hhs.gov" + df_link['link'] #add front part of
 ↪  link to make it clickable
```

Link: link=href class, under h2 class. 'h2 class' under 'header class' under 'div class' under 'li class'

```
### combine dataframes
df_Q1 = pd.concat([df_title, df_date, df_category, df_link], axis=1)
print(df_Q1.head())
```

```
                                               title              date  \
0  Boise Nurse Practitioner Sentenced To 48 Month...  November 7, 2024
1  Former Traveling Nurse Pleads Guilty To Tamper...  November 7, 2024
2  Former Arlington Resident Sentenced To Prison ...  November 7, 2024
```

```
3  Paroled Felon Sentenced To Six Years For Fraud...  November 7, 2024
4  Former Licensed Counselor Sentenced For Defrau...  November 6, 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/boise-nu...
1  https://oig.hhs.gov/fraud/enforcement/former-t...
2  https://oig.hhs.gov/fraud/enforcement/former-a...
3  https://oig.hhs.gov/fraud/enforcement/paroled-...
4  https://oig.hhs.gov/fraud/enforcement/former-l...
```

2. (Partner 1) Crawling: Then for each enforcement action, click the link and collect the name of the agency involved (e.g., for this link, it would be U.S. Attorney's Office, Eastern District of Washington).

```
### webcrawl
urls_1_2 = df_Q1['link']

### find agency by finding 'li' tags nested in 'article' tags
article_tags = []
for url in urls_1_2:
    response = requests.get(url)
    if response.status_code == 200:  # Check if the request was successful
        soup = BeautifulSoup(response.text, 'html.parser')

        # Find all <articel> tags
        #article_tags = soup2.find_all('article')

        # Find all <li> tags within <article> tags
        articles = soup.find_all('article')
        for article in articles:
            li_tags = article.find_all('li')
            for li in li_tags:
                article_tags.append(li.get_text(strip=True))  # Append the
 ↪  text of each <li> tag
    else:
```

```
        print(f"Failed to retrieve {link}")

### make dataframe and remove non-relevant items from list
df_agency = pd.DataFrame(article_tags)
df_agency.columns = ['agency']
df_agency = df_agency[df_agency['agency'].str.contains('Agency', case=False,
 ↪  na=False)]

print(df_agency.head())
#asked chat gpt 'how to go to every link in a list and extracts all 'li' tags
 ↪  that are nested in 'article' tag and create them into a new list'
```

```
                                               agency
1   Agency:November 7, 2024; U.S. Attorney's Offic...
5   Agency:U.S. Attorney's Office, District of Mas...
9   Agency:U.S. Attorney's Office, Eastern Distric...
13  Agency:U.S. Attorney's Office, Middle District...
17  Agency:U.S. Attorney's Office, Western Distric...
```

Name of agency is 'li' class, nested in 'ul' nested in 'div' nested in 'article'

*Some links do not have the corresponding agency listed

(30 points) Step 2: Making the scraper dynamic

1. Turning the scraper into a function: You will write a function that takes as input a month and a year, and then pulls and formats the enforcement actions like in Step 1 starting from that month+year to today.

a

b

c

(15 points) Step 3: Plot data based on scraped data (using altair)

1. (Partner 2) Plot a line chart that shows: the number of enforcement actions over time (aggregated to each month+year) overall since January 2021,

2. (Partner 1) Plot a line chart that shows: the number of enforcement actions split out by: ▪ "Criminal and Civil Actions" vs. "State Enforcement Agencies" ▪ Five topics in the "Criminal and Civil Actions" category: "Health Care Fraud", "Financial Fraud", "Drug Enforcement", "Bribery/Corruption", and "Other".

(15 points) Step 4: Create maps of enforcement activity For these questions, use this US Attorney District shapefile (link) and a Census state shapefile (link)

1. (Partner 1) Map by state: Among actions taken by state-level agencies, clean the state names you collected and plot a choropleth of the number of enforcement actions for each state. Hint: look for "State of" in the agency info!

2. (Partner 2) Map by district: Among actions taken by US Attorney District-level agencies, clean the district names so that you can merge them with the shapefile, and then plot a choropleth of the number of enforcement actions in each US Attorney District. Hint: look for "District" in the agency info.

(10 points) Extra credit: Calculate the enforcement actions on a per-capita basis (Both partners can work together)

1. Use the zip code shapefile from the previous problem set and merge it with zip code level population data. (Go to Census Data Portal, select "ZIP Code Tabulation Area", check "All 5-digit ZIP Code Tabulation Areas within United States", and under "P1 TOTAL POPULATION" select "2020: DEC Demographic and Housing Characteristics". Download the csv.).

2. Conduct a spatial join between zip code shapefile and the district shapefile, then aggregate to get population in each district.

3. Mapthe ratio of enforcement actions in each US Attorney District. You can calculate the ratio by aggregating the number of enforcement actions since January 2021 per district, and dividing it with the population data.