Problem Set 4

Rachel Alper

2001-02-07

Due 02/07 at 5:00PM Central.

"This submission is my work alone and complies with the 30538 integrity policy." Add your initials to indicate your agreement: RA

Github Classroom Assignment Setup and Submission Instructions

1. **Accepting and Setting up the PS4 Assignment Repository**
    - Each student must individually accept the repository for the problem set from Github Classroom ("ps4") – https://classroom.github.com/a/hWhtcHqH
        - You will be prompted to select your cnetid from the list in order to link your Github account to your cnetid.
        - If you can't find your cnetid in the link above, click "continue to next step" and accept the assignment, then add your name, cnetid, and Github account to this Google Sheet and we will manually link it: https://rb.gy/9u7fb6
    - If you authenticated and linked your Github account to your device, you should be able to clone your PS4 assignment repository locally.
    - Contents of PS4 assignment repository:
        - `ps4_template.qmd`: this is the Quarto file with the template for the problem set. You will write your answers to the problem set here.
2. **Submission Process**:
    - Knit your completed solution `ps4.qmd` as a pdf `ps4.pdf`.
        - Your submission does not need runnable code. Instead, you will tell us either what code you ran or what output you got.
    - To submit, push `ps4.qmd` and `ps4.pdf` to your PS4 assignment repository. Confirm on Github.com that your work was successfully pushed.

Grading

- You will be graded on what was last pushed to your PS4 assignment repository before the assignment deadline
- Problem sets will be graded for completion as: {missing (0%); - (incomplete, 50%); + (excellent, 100%)}
  - The percent values assigned to each problem denote how long we estimate the problem will take as a share of total time spent on the problem set, not the points they are associated with.
- In order for your submission to be considered complete, you need to push both your `ps4.qmd` and `ps4.pdf` to your repository. Submissions that do not include both files will automatically receive 50% credit.

```
import pandas as pd
import altair as alt
import time
import requests
from bs4 import BeautifulSoup

import warnings
warnings.filterwarnings('ignore')
alt.renderers.enable("png", ppi=200)
```

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

Step 1: Develop initial scraper and crawler

```
with open(r'hhs_website.html', 'r') as page:
    hhs_website = page.read()
    soup = BeautifulSoup(hhs_website, 'lxml')
```

```
records = []

cards = soup.find_all("li", class_="usa-card")

for card in cards:

    title = card.find("h2").get_text(strip=True)

    date = card.find("span").get_text(strip=True)

    category_tag = card.find("li", class_="usa-tag")
    category = category_tag.get_text(strip=True) if category_tag else None

    records.append({
        "title": title,
        "date": date,
        "category": category
    })


df = pd.DataFrame(records)
df["date"] = pd.to_datetime(df["date"])
```

```
df.head()
```

	title	date	category
0	Brooklyn Banker Pleads Guilty to Laundering Pr...	2026-02-03	COVID-19
1	Delafield Man Sentenced to 18 Months' Imprison...	2026-02-03	Criminal and Civil Actions
2	Former NFL Player Convicted for $197M Medicare...	2026-02-03	Criminal and Civil Actions
3	AG's Office Secures Indictments Against Peabod...	2026-02-02	State Enforcement Agencies
4	Florida Man Pleads Guilty to Conspiracy to Vio...	2026-01-30	Criminal and Civil Actions

Step 2: Making the scraper dynamic

1. Turning the scraper into a function

- a. Pseudo-Code

Step 1: Validate the input and ensure that the start year is not prior to 2013 and that the indicator to run the function is turned on.

Step 2: Initialize variables including base url, page number, records, and compute a start date.

Step 3: Use a while loop (this is better than a for loop because we don't know how many pages we need; we just don't want to go earlier than the start date). The while loop will take the following form:

while True: scrape page extract titles, dates, categories

```
if oldest date on page < start_date:
    break

page += 1
sleep(1)
```

Step 4: Save to date time format and convert to csv

- b. Create Dynamic Scraper

```
def scrape_enforcement_actions(start_month, start_year, run_scraper=True):

    if not run_scraper:
        print("Scraper is turned off.")
        return None

    if start_year < 2013:
        print("Must have a start year >= 2013")
        return None


    start_date = pd.Timestamp(year=start_year, month=start_month, day=1)

    base_url = "https://oig.hhs.gov/fraud/enforcement/"
    page = 1
    records = []

    while True:

        url = base_url if page == 1 else f"{base_url}?page={page}"
        print(f"Scraping page {page}...")

        response = requests.get(url)
        soup = BeautifulSoup(response.text, "lxml")

        cards = soup.find_all("li", class_="usa-card")

        if not cards:
            break

        page_dates = []

        for card in cards:

            title = card.find("h2").get_text(strip=True)
            date_text = card.find("span").get_text(strip=True)
            date = pd.to_datetime(date_text)

            category_tags = card.find_all("li", class_="usa-tag")
            categories = ", ".join(tag.get_text(strip=True) for tag in
↪   category_tags)

            page_dates.append(date)
```

```
            records.append({
                "title": title,
                "date": date,
                "category": categories
            })

        if min(page_dates) < start_date:
            break

        page += 1
        time.sleep(1)


    df = pd.DataFrame(records)

    df = df[df["date"] >= start_date].sort_values("date", ascending=False)

    filename = f"enforcement_actions_{start_year}_{start_month}.csv"
    df.to_csv(filename, index=False)

    print(f"\nSaved to {filename}")
    print(f"Total enforcement actions: {len(df)}")

    return df
```

Now we can run the function for January 2024 and then change the indicator to false:

```
df_2024 = scrape_enforcement_actions(1, 2024, run_scraper=False)
```

```
Scraper is turned off.
```

```
file_path = 'enforcement_actions_2024_1.csv'
df_2024 = pd.read_csv(file_path, index_col = 0)

print("The number of enforcement actions found is {}".format(len(df_2024)))
print("The earliest date in the datat is
 ↪  {}".format(df_2024.sort_values("date").iloc[0]))
```

```
The number of enforcement actions found is 1769
The earliest date in the datat is date                          2024-01-03
```

```
category    State Enforcement Agencies
Name: Former Nurse Aide Indicted In Death Of Clarksville Patient Arrested In
Georgia, dtype: object
```

- c. Test Your Code

```
df_2022 = scrape_enforcement_actions(1, 2022, run_scraper=False)
```

```
Scraper is turned off.
```

```
file_path = 'enforcement_actions_2022_1.csv'
df = pd.read_csv(file_path, index_col = 0)
df = df.reset_index()

print("The number of enforcement actions found is {}".format(len(df)))
print("The earliest date in the datat is
 ↪  {}".format(df.sort_values("date").iloc[0]))
```

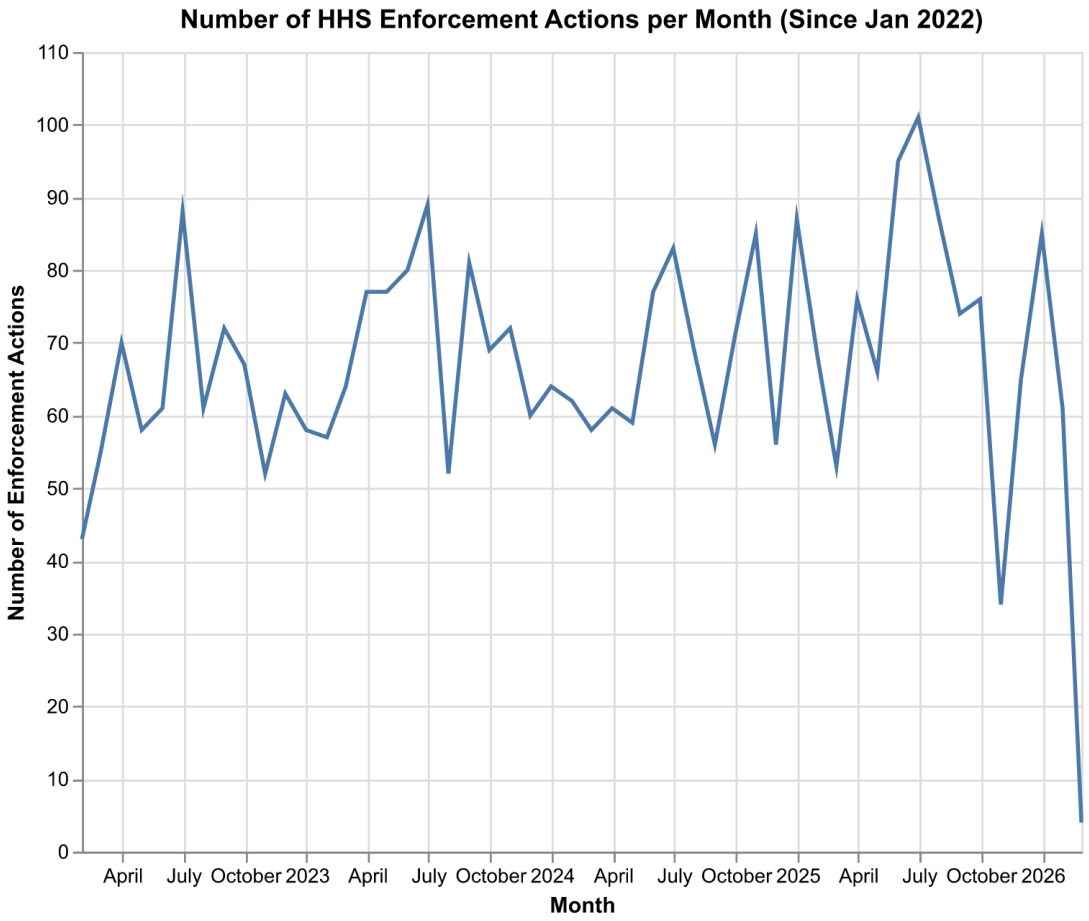
```
The number of enforcement actions found is 3359
The earliest date in the datat is title       Integrated Pain Management
Medical Group Agree...
date                                                2022-01-04
category                                 Fraud Self-Disclosures
Name: 3358, dtype: object
```

Step 3: Plot data based on scraped data

1. Plot the number of enforcement actions over time

```
df["date"] = pd.to_datetime(df["date"], errors="coerce")

monthly_counts = (
    df
    .set_index("date")
    .resample("M")
    .size()
    .reset_index(name="num_actions")
)

chart = alt.Chart(monthly_counts).mark_line().encode(
    x=alt.X("date:T", title="Month"),
```

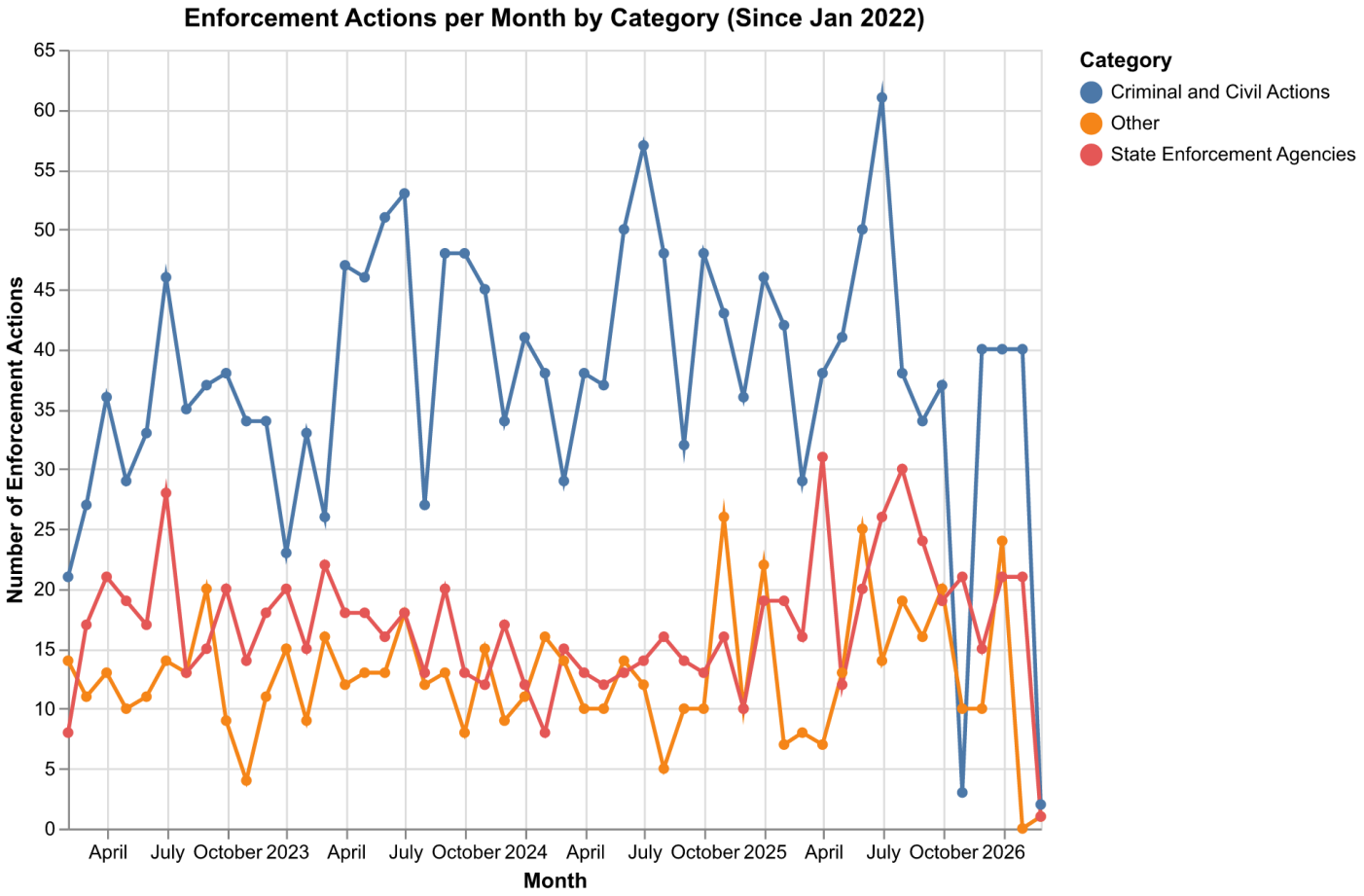
```
    y=alt.Y("num_actions:Q", title="Number of Enforcement Actions"),
    tooltip=["date:T", "num_actions:Q"]
).properties(
    title="Number of HHS Enforcement Actions per Month (Since Jan 2022)",
    width=500,
    height=400
)

chart
```



2. Plot the number of enforcement actions categorized:

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

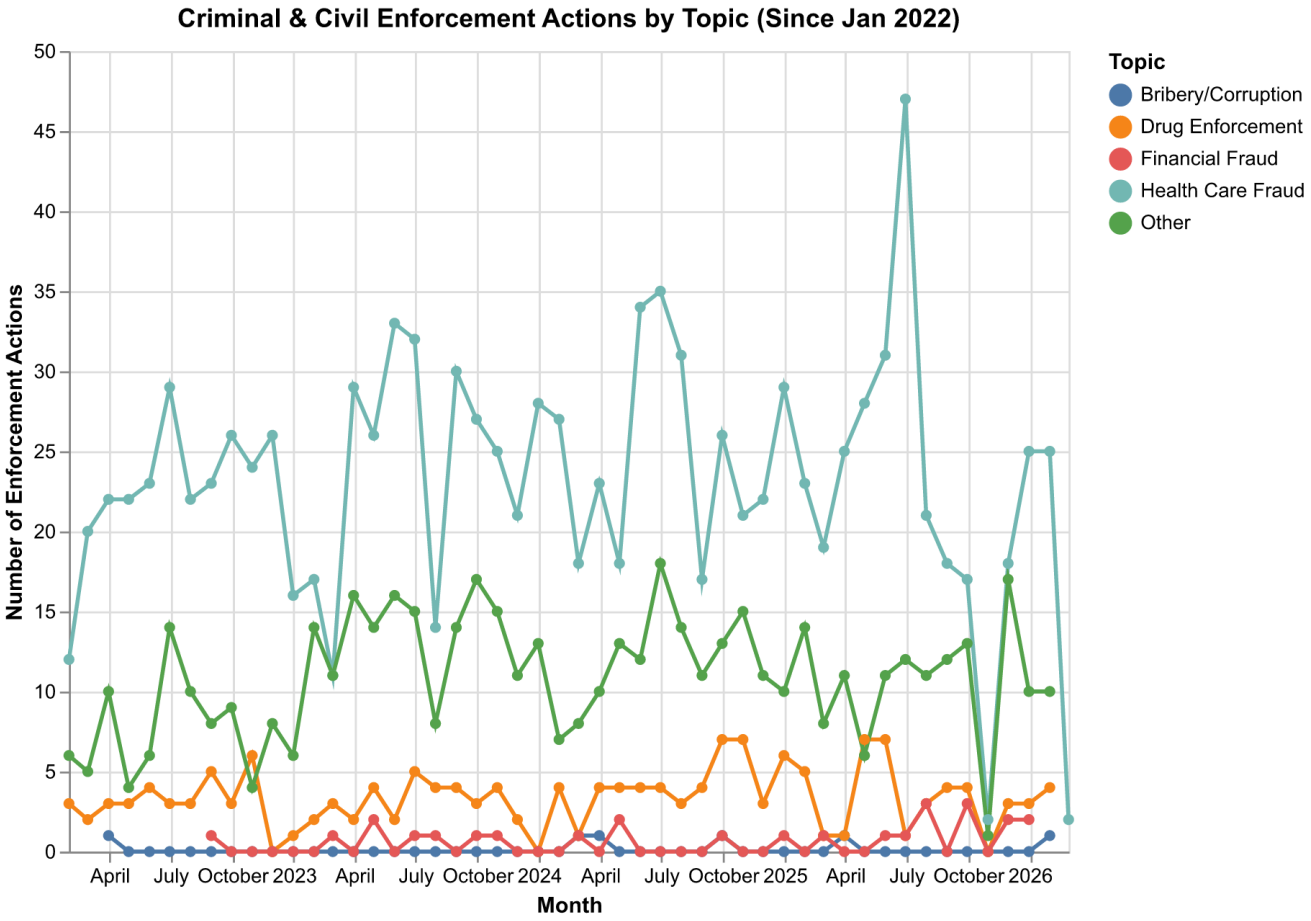
```
def simplify_category(cat):
    if "Criminal and Civil Actions" in cat:
        return "Criminal and Civil Actions"
    elif "State Enforcement Agencies" in cat:
        return "State Enforcement Agencies"
    else:
        return "Other"

df["main_category"] = df["category"].apply(simplify_category)

monthly_cat = (
    df
    .set_index("date")
    .groupby("main_category")
    .resample("M")
    .size()
    .reset_index(name="num_actions")
)

alt.Chart(monthly_cat).mark_line(point=True).encode(
    x=alt.X("date:T", title="Month"),
    y=alt.Y("num_actions:Q", title="Number of Enforcement Actions"),
    color=alt.Color("main_category:N", title="Category"),
    tooltip=["date:T", "main_category:N", "num_actions:Q"]
).properties(
    title="Enforcement Actions per Month by Category (Since Jan 2022)",
    width=500,
    height=400
)
```



- based on five topics

```
import re

def classify_topic(title):
    title = title.lower()

    if
     ↪ re.search(r"health|medicare|medicaid|kickback|patient|clinic|chiropractor|hospital",
     ↪ title):
        return "Health Care Fraud"

    elif re.search(r"bank|financial|loan|money laundering|wire
     ↪ fraud|investment", title):
        return "Financial Fraud"
```

```
    elif re.search(r"drug|opioid|controlled substance|pharmacy|fentanyl",
     ↪ title):
        return "Drug Enforcement"

    elif re.search(r"brib|corrupt|kickback scheme", title):
        return "Bribery/Corruption"

    else:
        return "Other"

criminal_df = df[df["main_category"] == "Criminal and Civil Actions"].copy()
criminal_df["topic"] = criminal_df["title"].apply(classify_topic)

monthly_topics = (
    criminal_df
    .set_index("date")
    .groupby("topic")
    .resample("M")
    .size()
    .reset_index(name="num_actions")
)

alt.Chart(monthly_topics).mark_line(point=True).encode(
    x=alt.X("date:T", title="Month"),
    y=alt.Y("num_actions:Q", title="Number of Enforcement Actions"),
    color=alt.Color("topic:N", title="Topic"),
    tooltip=["date:T", "topic:N", "num_actions:Q"]
).properties(
    title="Criminal & Civil Enforcement Actions by Topic (Since Jan 2022)",
    width=500,
    height=400
)
```


Criminal & Civil Enforcement Actions by Topic (Since Jan 2022)
Number of Enforcement Actions
50
45
40
35
30
25
20
15
10
5
0
April
July
October
2023
April
July
October
2024
April
July
October
2025
April
July
October
2026
Month
Topic
Bribery/Corruption
Drug Enforcement
Financial Fraud
Health Care Fraud
Other