PS4

Jessica Prus

2026-02-06

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: JP

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.

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

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

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

Step 1: Develop initial scraper and crawler

```python
# scrape first page

# download and save html
url = 'https://oig.hhs.gov/fraud/enforcement/'
response = requests.get(url)

# convert into beautifulsoup object
soup = BeautifulSoup(response.text, 'lxml')
soup.text[0:50]
```

```
'\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\nEnforcement Actions | Office '
```

```python
# scrape first page

# first pass to extract information
tag = soup.find_all('li')

# sanity check
len(tag)
```

```
145
```

```python
# scrape first page

actions = soup.find_all('li', class_='usa-card card--list pep-card--minimal
 ↪  mobile:grid-col-12')

# initialize empty list
data = []

# add to data list
for li in actions:
  # title + link
  a_tag = li.find('h2', class_='usa-card__heading').find('a')
  title = a_tag.get_text(strip=True)
  link = 'https://oig.hhs.gov' + a_tag['href']

  # date
  date = li.find('span').get_text(strip=True)

  # category
  category = li.find('li', class_='usa-tag').get_text(strip=True)

  data.append({
    'title': title,
    'date': date,
    'category': category,
    'link': link
  })

# view results
for row in data:
  print(row)
```

```
{'title': 'Houston Transplant Doctor Indicted For Making False Statements In
Patients’ Medical Records', 'date': 'February 5, 2026', 'category': 'Criminal
and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/houston-transplant-doctor-indicted-for-making-false-s
{'title': 'MultiCare Health System to Pay Millions to Settle Fraud Case',
'date': 'February 4, 2026', 'category': 'Criminal and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/multicare-health-system-to-pay-millions-to-settle-frau
{'title': 'Brooklyn Banker Pleads Guilty to Laundering Proceeds of Medicare
Fraud for Transnational Criminal Organization', 'date': 'February 3, 2026',
'category': 'COVID-19', 'link':
'https://oig.hhs.gov/fraud/enforcement/brooklyn-banker-pleads-guilty-to-laundering-proceeds-o
```

```
{'title': 'Delafield Man Sentenced to 18 Months’ Imprisonment for Conspiracy
to Pay Health Care Kickbacks', 'date': 'February 3, 2026', 'category':
'Criminal and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/delafield-man-sentenced-to-18-months-imprisonment-for-
{'title': 'Former NFL Player Convicted for $197M Medicare Fraud', 'date':
'February 3, 2026', 'category': 'Criminal and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/former-nfl-player-convicted-for-197m-medicare-fraud/'}
{'title': 'Attorney General Hanaway Obtains Medicaid Fraud Conviction Against
Couple Lying About Marital Status To Steal Funds', 'date': 'February 3,
2026', 'category': 'State Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/attorney-general-hanaway-obtains-medicaid-fraud-convic
{'title': "AG's Office Secures Indictments Against Peabody Alcohol and Drug
Counselor and Her Businesses for More Than $850,000 in MassHealth Fraud",
'date': 'February 2, 2026', 'category': 'State Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/ags-office-secures-indictments-against-peabody-alcohol
{'title': 'Florida Man Pleads Guilty to Conspiracy to Violate the
Anti-Kickback Statute', 'date': 'January 30, 2026', 'category': 'Criminal and
Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/florida-man-pleads-guilty-to-conspiracy-to-violate-the
{'title': 'Forefront Living Hospice Agreed to Pay $1.9 Million for Allegedly
Violating the Civil Monetary Penalties Law by Submitting Claims for Services
that Identified the Incorrect Provider or Were Performed by Non-enrolled or
Incorrect Providers', 'date': 'January 30, 2026', 'category': 'Fraud
Self-Disclosures', 'link':
'https://oig.hhs.gov/fraud/enforcement/forefront-living-hospice-agreed-to-pay-19-million-for-
{'title': 'Attorney General Jeff Jackson Announces Health Care Fraud
Conviction and Settlement', 'date': 'January 30, 2026', 'category': 'State
Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/attorney-general-jeff-jackson-announces-health-care-f
{'title': 'Yadkinville Woman Sentenced in Connection with Multi-Million
Dollar Medicaid Fraud Scheme', 'date': 'January 29, 2026', 'category':
'Criminal and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/yadkinville-woman-sentenced-in-connection-with-multi-m
{'title': 'Attorney General Labrador Announces Sentencing of Kootenai County
Woman for Public Assistance Provider Fraud', 'date': 'January 29, 2026',
'category': 'State Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/attorney-general-labrador-announces-sentencing-of-koo
{'title': 'Attorney General Hanaway Obtains Medicaid Fraud Conviction For
Services Not Provided', 'date': 'January 29, 2026', 'category': 'State
Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/attorney-general-hanaway-obtains-medicaid-fraud-convic
```

```
{'title': 'Holmes Regional Medical Center Agreed to Pay $113,000 for
Allegedly Violating Patient Dumping Statute by Failing to Provide an
Appropriate Medical Screening Examination', 'date': 'January 28, 2026',
'category': 'CMP and Affirmative Exclusions', 'link':
'https://oig.hhs.gov/fraud/enforcement/holmes-regional-medical-center-agreed-to-pay-113000-fo
{'title': 'Slidell Chiropractor Sentenced for Health Care Fraud', 'date':
'January 28, 2026', 'category': 'COVID-19', 'link':
'https://oig.hhs.gov/fraud/enforcement/slidell-chiropractor-sentenced-for-health-care-fraud/
{'title': 'Repeat Health Care Fraud Offender Sentenced for Defrauding New
Hampshire Medicaid', 'date': 'January 28, 2026', 'category': 'Criminal and
Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/repeat-health-care-fraud-offender-sentenced-for-defrau
{'title': 'Scranton Heart Institute Agrees To Pay $48,709.20 To Settle False
Claims Act Allegations', 'date': 'January 28, 2026', 'category': 'Criminal
and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/scranton-heart-institute-agrees-to-pay-4870920-to-sett
{'title': 'Rheumatologist Agrees To Resolve False Claims Act Allegations
Related To Unapproved Drugs', 'date': 'January 28, 2026', 'category':
'Criminal and Civil Actions', 'link':
'https://oig.hhs.gov/fraud/enforcement/rheumatologist-agrees-to-resolve-false-claims-act-alle
{'title': 'Attorney General James Uthmeier Announces Arrests in Central
Florida Medicaid Fraud Scheme', 'date': 'January 28, 2026', 'category':
'State Enforcement Agencies', 'link':
'https://oig.hhs.gov/fraud/enforcement/attorney-general-james-uthmeier-announces-arrests-in-c
{'title': 'Cordell Memorial Hospital Agreed to Pay $40,000 for Allegedly
Violating Patient Dumping Statute by Failing to Provide an Appropriate
Medical Screening Examination', 'date': 'January 27, 2026', 'category': 'CMP
and Affirmative Exclusions', 'link':
'https://oig.hhs.gov/fraud/enforcement/cordell-memorial-hospital-agreed-to-pay-40000-for-alle
```

```python
# scrape first page

# convert into dataframe
df = pd.DataFrame(data)

# print the head of the dataframe
print(df.head)
```

```
<bound method NDFrame.head of
title              date  \
0   Houston Transplant Doctor Indicted For Making ...  February 5, 2026
1   MultiCare Health System to Pay Millions to Set...  February 4, 2026
2   Brooklyn Banker Pleads Guilty to Laundering Pr...  February 3, 2026
```

```
3   Delafield Man Sentenced to 18 Months’ Imprison...  February 3, 2026
4   Former NFL Player Convicted for $197M Medicare...  February 3, 2026
5   Attorney General Hanaway Obtains Medicaid Frau...  February 3, 2026
6   AG's Office Secures Indictments Against Peabod...  February 2, 2026
7   Florida Man Pleads Guilty to Conspiracy to Vio...  January 30, 2026
8   Forefront Living Hospice Agreed to Pay $1.9 Mi...  January 30, 2026
9   Attorney General Jeff Jackson Announces Health...  January 30, 2026
10  Yadkinville Woman Sentenced in Connection with...  January 29, 2026
11  Attorney General Labrador Announces Sentencing...  January 29, 2026
12  Attorney General Hanaway Obtains Medicaid Frau...  January 29, 2026
13  Holmes Regional Medical Center Agreed to Pay $...  January 28, 2026
14  Slidell Chiropractor Sentenced for Health Care...  January 28, 2026
15  Repeat Health Care Fraud Offender Sentenced fo...  January 28, 2026
16  Scranton Heart Institute Agrees To Pay $48,709...  January 28, 2026
17  Rheumatologist Agrees To Resolve False Claims ...  January 28, 2026
18  Attorney General James Uthmeier Announces Arre...  January 28, 2026
19  Cordell Memorial Hospital Agreed to Pay $40,00...  January 27, 2026

                          category  \
0       Criminal and Civil Actions
1       Criminal and Civil Actions
2                         COVID-19
3       Criminal and Civil Actions
4       Criminal and Civil Actions
5       State Enforcement Agencies
6       State Enforcement Agencies
7       Criminal and Civil Actions
8           Fraud Self-Disclosures
9       State Enforcement Agencies
10      Criminal and Civil Actions
11      State Enforcement Agencies
12      State Enforcement Agencies
13  CMP and Affirmative Exclusions
14                        COVID-19
15      Criminal and Civil Actions
16      Criminal and Civil Actions
17      Criminal and Civil Actions
18      State Enforcement Agencies
19  CMP and Affirmative Exclusions

                                                 link
0   https://oig.hhs.gov/fraud/enforcement/houston-...
1   https://oig.hhs.gov/fraud/enforcement/multicar...
```

```
2   https://oig.hhs.gov/fraud/enforcement/brooklyn...
3   https://oig.hhs.gov/fraud/enforcement/delafiel...
4   https://oig.hhs.gov/fraud/enforcement/former-n...
5   https://oig.hhs.gov/fraud/enforcement/attorney...
6   https://oig.hhs.gov/fraud/enforcement/ags-offi...
7   https://oig.hhs.gov/fraud/enforcement/florida-...
8   https://oig.hhs.gov/fraud/enforcement/forefron...
9   https://oig.hhs.gov/fraud/enforcement/attorney...
10  https://oig.hhs.gov/fraud/enforcement/yadkinvi...
11  https://oig.hhs.gov/fraud/enforcement/attorney...
12  https://oig.hhs.gov/fraud/enforcement/attorney...
13  https://oig.hhs.gov/fraud/enforcement/holmes-r...
14  https://oig.hhs.gov/fraud/enforcement/slidell-...
15  https://oig.hhs.gov/fraud/enforcement/repeat-h...
16  https://oig.hhs.gov/fraud/enforcement/scranton...
17  https://oig.hhs.gov/fraud/enforcement/rheumato...
18  https://oig.hhs.gov/fraud/enforcement/attorney...
19  https://oig.hhs.gov/fraud/enforcement/cordell-...  >
```

Step 2: Making the scraper dynamic

1. Turning the scraper into a function

- a. Pseudo-Code

create knit_indicator <- create an indicator to knit

FUNCTION scraper_function (month, year, run): <- define scraper as a function BEGIN

run_indicator <- create running indicator print(not run) <- print that the scraper is not run return none

if year 2013 <- check year print(low year) <- print that that year is under 2013 return none

start_date <- set start date

base_url <- set base url data_list <- create an empty list to hold data

page_1 <- set page as 1 keep_scraper <- set scraping indicator to true

while_scraping: <- set WHILE statement for scraping

```
page_url <- set the url of the page
response_page <- get information from the page
soup <- scrape text using BeautifulSoup
```

```
find_actions <- find actions using BeautifulSoup find_all
  find_li <- find li tags
  action_class <- find tag li with class of the actions

if_not_actions: <- if actions is not desired tag or class
  break

for_li_actions: <- if li tag is of an enforcement action
  li_heading <- look for headings of actions
  a_tag <- when action has specific heading then return the lowest substring

  action_title <- scrape title
  action_link <- scrape link

  date_tag <- look for the tag which marks the date
  action_date <- scrape date

  category_tag <- look for the tag which marks the category
  action_category <- scrape category
  date_datetime <- convert date to datetime

  stop_condition: <- if year < 2013:
    no_scrape <- don't scrape
    break

  csv_list_append <- append to csv list
    set_title <- set action column
    set_date <- set action date
    set_cat <- set action category
    set_link <- set action link

print_page <- print the page number after finished scraping
page_add <- increase page number by 1
time_sleep <- set one second delay between scraping pages
```

convert_dataframe <- convert list to dataframe

convert_csv <- convert dataframe to csv

confirm_csv <- print that the csv has been created return dataframe

END ENDFUNCTION

- b. Create Dynamic Scraper

```python
# create scraper as a function
def scrape_from(month, year, run=False):

  # indicator to prevent re-running when knitting
  if not run:
    print('Scraper not run. Set run=True to generate the CSV')
    return None

  # check year
  if year < 2013:
    print('Please restrict to year >= 2013')
    return None

  # set start date
  start_date = pd.Timestamp(year=year, month=month, day=1)

  # initialize base information
  base_url = 'https://oig.hhs.gov/fraud/enforcement/'
  all_data = []

  page =1
  keep_scraping = True

  # create scraper
  while keep_scraping:

    url = f'{base_url}?page={page}'
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'lxml')

    actions = soup.find_all(
      'li',
      class_='usa-card card--list pep-card--minimal mobile:grid-col-12'
    )

    if not actions:
      break

    for li in actions:
      heading = li.find('h2', class_='usa-card__heading')
      a_tag = heading.find('a') if heading else None

      title = a_tag.get_text(strip=True) if a_tag else None
      link = 'https://oig.hhs.gov' + a_tag['href'] if a_tag else None

      date_tag = li.find('span')
      date_text = date_tag.get_text(strip=True) if date_tag else None
      date = pd.to_datetime(date_text)

      cat_tag = li.find('li', class_='usa-tag')
      category = cat_tag.get_text(strip=True) if cat_tag else None

      # stop condition
      if date < start_date:
        keep_scraping = False
        break

      # append to list
      all_data.append({
        'title': title,
        'date': date,
        'category': category,
        'link': link
      })

    print(f'Finished page {page}')
    page += 1
    time.sleep(1) # required delay

  # convert to dataframe
  df_actions = pd.DataFrame(all_data)

  # save to csv
  df_actions.to_csv('enforcement_actions_year_month.csv', index=False)

  print('Saved enforcement_actions_year_month.csv')
  return df_actions

# create indicator for knitting
scrape_from(month=1, year=2024, run=False)
```

```
Scraper not run. Set run=True to generate the CSV
```

```
# load csv
df_actions = pd.read_csv('enforcement_actions_year_month.csv')
```

```
df_actions['date'] = pd.to_datetime(df_actions['date'])
```

```
# sort and print earliest row
earliest = df_actions.sort_values('date').iloc[0]
print(earliest)
```

```
title       Integrated Pain Management Medical Group Agree...
date                                      2022-01-04 00:00:00
category                              Fraud Self-Disclosures
link        https://oig.hhs.gov/fraud/enforcement/integrat...
Name: 3376, dtype: object
```

1787 enforcement actions are in the final dataframe. The date of the earliest enforcement action is 2024-01-03. It's title is "Former Nurse Aide Indicted In Death Of Clarksville Patient Arrested in Georgia" and the category is State Enforcement Agencies.

- c. Test Your Code

```
# create scraper as a function
def scrape_from(month, year, run=False):

  # indicator to prevent re-running when knitting
  if not run:
    print('Scraper not run. Set run=True to generate the CSV')
    return None

  # check year
  if year < 2013:
    print('Please restrict to year >= 2013')
    return None

  # set start date
  start_date = pd.Timestamp(year=year, month=month, day=1)

  # initialize base information
  base_url = 'https://oig.hhs.gov/fraud/enforcement/'
  all_data = []

  page =1
  keep_scraping = True

  # create scraper
```

```python
while keep_scraping:

  url = f'{base_url}?page={page}'
  response = requests.get(url)
  soup = BeautifulSoup(response.text, 'lxml')

  actions = soup.find_all(
    'li',
    class_='usa-card card--list pep-card--minimal mobile:grid-col-12'
  )

  if not actions:
    break

  for li in actions:
    heading = li.find('h2', class_='usa-card__heading')
    a_tag = heading.find('a') if heading else None

    title = a_tag.get_text(strip=True) if a_tag else None
    link = 'https://oig.hhs.gov' + a_tag['href'] if a_tag else None

    date_tag = li.find('span')
    date_text = date_tag.get_text(strip=True) if date_tag else None
    date = pd.to_datetime(date_text)

    cat_tag = li.find('li', class_='usa-tag')
    category = cat_tag.get_text(strip=True) if cat_tag else None

    # stop condition
    if date < start_date:
      keep_scraping = False
      break

    # append to list
    all_data.append({
      'title': title,
      'date': date,
      'category': category,
      'link': link
    })

  print(f'Finished page {page}')
  page += 1
```

```
    time.sleep(1) # required delay

  # convert to dataframe
  df_actions = pd.DataFrame(all_data)

  # save to csv
  df_actions.to_csv('enforcement_actions_year_month.csv', index=False)

  print('Saved enforcement_actions_year_month.csv')
  return df_actions

# create indicator for knitting
scrape_from(month=1, year=2022, run=False)
```

```
Scraper not run. Set run=True to generate the CSV
```

```
# load csv
df_actions = pd.read_csv('enforcement_actions_year_month.csv')
df_actions['date'] = pd.to_datetime(df_actions['date'])
```

```
# sort and print earliest row
earliest = df_actions.sort_values('date').iloc[0]
print(earliest)
```

```
title       Integrated Pain Management Medical Group Agree...
date                                      2022-01-04 00:00:00
category                               Fraud Self-Disclosures
link        https://oig.hhs.gov/fraud/enforcement/integrat...
Name: 3376, dtype: object
```

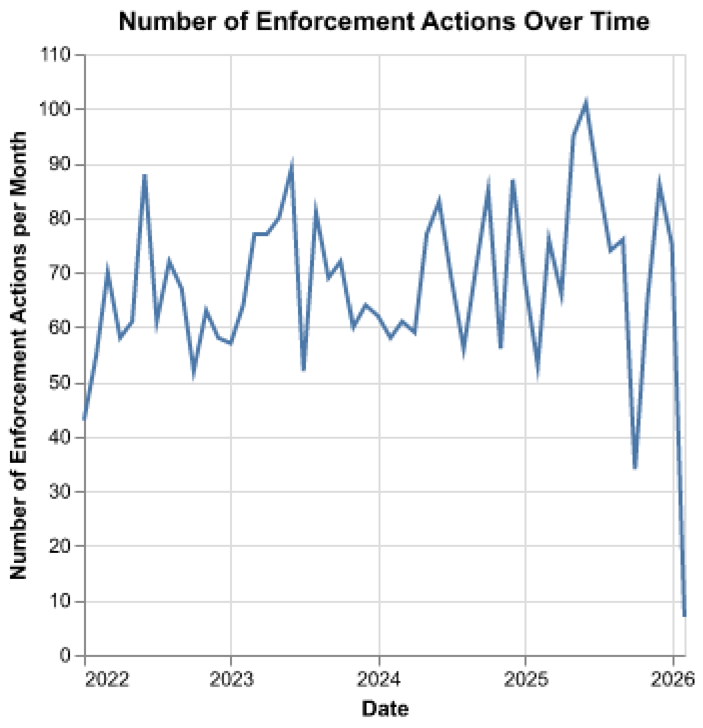
There are 3377 enforcement actions in the final dataframe. The earliest enforcement action it scaped is from 2022-01-04. Its title is "Integrated Pain Management Group Agreed to Pay $10,000 for Allegedly Violating the Civil Monetary Penalties Law by Employing Excluded Individuals" and its category is Fraud Self Disclosures.

Step 3: Plot data based on scraped data

1. Plot the number of enforcement actions over time

```
# aggregate to month and year
df_actions['yearmonth'] =df_actions['date'].dt.to_period('M').astype(str)
df_monthly = df_actions.groupby('yearmonth').size().reset_index(name='count')

# create the plot
chart = alt.Chart(df_monthly).mark_line().encode(
  x=alt.X('yearmonth:T', title='Date', axis=alt.Axis(format='%Y')),
  y=alt.Y('count:Q', title='Number of Enforcement Actions per Month')
).properties(
  title='Number of Enforcement Actions Over Time'
)
chart
```



2. Plot the number of enforcement actions categorized:

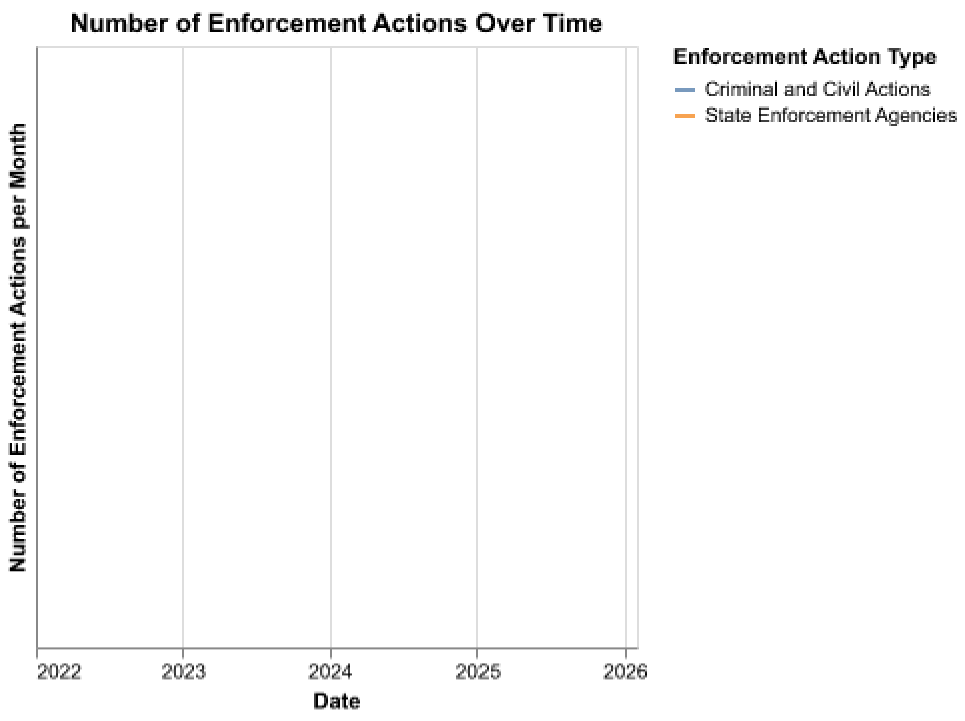
- based on “Criminal and Civil Actions” vs. “State Enforcement Agencies”

```
# change yearmonth to timestamp
df_actions['yearmonth'] = df_actions['date'].dt.to_period('M').astype(str)

# filter dataframe categories
filtered_df = df_actions[df_actions['category'].isin([
  'Criminal and Civil Actions',
  'State Enforcement Agencies'
])]

# aggregate by month and category
df_monthly = (
  filtered_df
  .groupby(['yearmonth', 'category'])
  .size()
  .reset_index(name='count')
)

# create the plot
chart = alt.Chart(filtered_df).mark_line().encode(
  x=alt.X('yearmonth:T', title='Date'),
  y=alt.Y('count:Q', title='Number of Enforcement Actions per Month'),
  color=alt.Color('category:N',
  legend=alt.Legend(title='Enforcement Action Type'))
).properties(
  title='Number of Enforcement Actions Over Time'
)
chart
```



- based on five topics

```
# filter to criminal and civil actoins
df_cc = df_actions[df_actions['category'] == 'Criminal and Civil
↪  Actions'].copy()

# set keywords
topic_keywords = {
  'Health Care Fraud': ['medicare', 'medicaid', 'heath', 'hospital',
   ↪  'physician','nurse',
  'medical', 'provider'],
  'Financial Fraud': ['bank', 'loan', 'financial', 'investment', 'finance'],
  'Drug Enforcement': ['drug', 'opioid', 'pharmacy', 'pill'],
  'Bribery/Corruption': ['bribe', 'corruption', 'kickback']
}

def classify_topic(title):
  t = title.lower()
  for topic, words in topic_keywords.items():
    if any(w in t for w in words):
      return topic
  return 'Other'
```

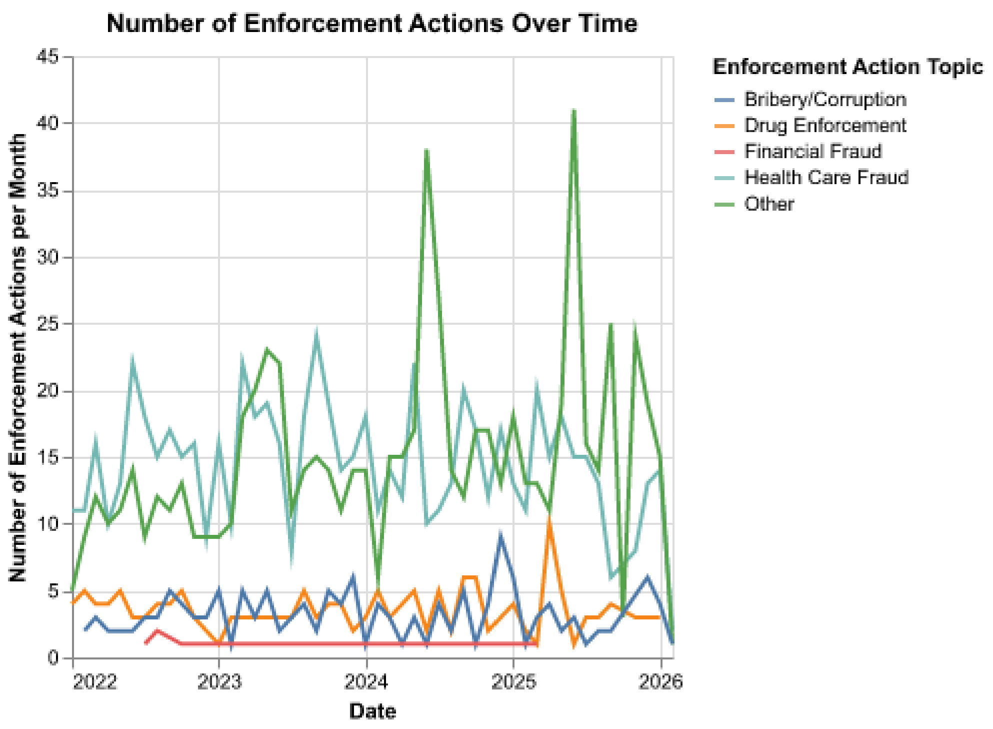
```
# apply classifier to new column
df_cc['topic'] = df_cc['title'].apply(classify_topic)

# sanity check
df_cc['topic'].value_counts()
df_cc[['title', 'topic']].sample(10)

# aggregate by month and topic
df_cc['yearmonth'] = df_cc['date'].dt.to_period('M').dt.to_timestamp()

df_monthly = (
  df_cc
  .groupby(['yearmonth', 'topic'])
  .size()
  .reset_index(name='count')
)

# create the plot
chart = alt.Chart(df_monthly).mark_line().encode(
  x=alt.X('yearmonth:T', title='Date', axis=alt.Axis(format='%Y')),
  y=alt.Y('count:Q', title='Number of Enforcement Actions per Month'),
  color=alt.Color('topic:N',
  legend=alt.Legend(title='Enforcement Action Topic'))
).properties(
  title='Number of Enforcement Actions Over Time'
)
chart
```


Number of Enforcement Actions Over Time
Number of Enforcement Actions per Month
45
40
35
30
25
20
15
10
5
0
2022
2023
2024
2025
2026
Date
Enforcement Action Topic
Bribery/Corruption
Drug Enforcement
Financial Fraud
Health Care Fraud
Other