quarto preview /Users/suyuanfang/Desktop/Pyhton/ps5/ps5_template.qmd –no-browser –no-watch-inputs— title: “title” author: “author” date: “date” format: html: default pdf: include-in-header: text: | include-before-body: text: | output: echo: false eval: false

---

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 (name and cnet ID): Suyuan Fang - suyuanfang
    - Partner 2 (name and cnet ID): Jiaxuan nie - Jnie21
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: SF JN
5. “I have uploaded the names of anyone else other than my partner and I worked with on the problem set **yes**” (1 point)
6. Late coins used this pset:1 Late coins left after submission:1
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')
```

Step 1: Develop initial scraper and crawler

1. Scraping (PARTNER 1)

```
import requests
from bs4 import BeautifulSoup
import pandas as pd

url = "https://oig.hhs.gov/fraud/enforcement/"
response = requests.get(url)
```

```
soup = BeautifulSoup(response.text, 'html.parser')

data = []

for action in soup.select("li.usa-card"):

    title = action.select_one("h2.usa-card__heading a").get_text(strip=True)

    link = "https://oig.hhs.gov" + action.select_one("h2.usa-card__heading a")["href"]

    date = action.select_one("span.text-base-dark.padding-right-105").get_text(strip=T

    category = action.select_one("ul.display-inline.add-list-reset li").get_text(strip

    data.append({
        "Title": title,
        "Date": date,
        "Category": category,
        "Link": link
    })

df = pd.DataFrame(data)
print(df.head())
save_path = "/Users/suyuanfang/Desktop/Pyhton/ps5/enforcement_actions.csv"
df.to_csv(save_path, index=False)
```

```
                                               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)

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

```
data_update = []

for action in soup.select("li.usa-card"):
    title = action.select_one("h2.usa-card__heading a").get_text(strip=True)

    link = "https://oig.hhs.gov" + action.select_one("h2.usa-card__heading a")["href"]

    date = action.select_one("span.text-base-dark.padding-right-105").get_text(strip=T

    category = action.select_one("ul.display-inline.add-list-reset li").get_text(strip

    detail_response = requests.get(link)
    detail_soup = BeautifulSoup(detail_response.text, 'html.parser')

    spans = detail_soup.select("span.padding-right-2.text-base")
    agency = spans[1].find_parent("li").get_text(strip=True).replace("Agency:", "").st

    data_update.append({
        "Title": title,
        "Date": date,
        "Category": category,
        "Link": link,
        "Agency": agency
    })

df = pd.DataFrame(data_update)
print(df.head())
save_path = "/Users/suyuanfang/Desktop/Pyhton/ps5/enforcement_actions_update.csv"
df.to_csv(save_path, index=False)
```

```
                                               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) Function scrape_enforcement_actions(start_date): Initialize an empty list to store results Set the current page number to 1 Set base URL for the enforcement actions page

  WHILE True: Construct the URL for the current page using the base URL and the page number Make a request to the URL and retrieve the HTML content Parse the HTML content to find enforcement action entries

  ```
    IF there are no enforcement action entries on this page:
        Break the loop (no more pages to scrape)

    FOR each action entry in the page:
        Extract the title, link, date, and category
        Convert the date of the action to a datetime object

        IF action date is older than start_date:
            Continue to the next page (skip this entry if it's before the target
  date)

        Make a request to the detail page using the link to get additional
  information
        Parse the detail page to find the agency name
        Append all extracted data (title, date, category, link, agency) to results
  list

    Wait 1 second to avoid too frequent requests
    Increment the page number to go to the next page
  ```

  Convert results list to a DataFrame Save DataFrame to a CSV file with the start date in the file name Print a message indicating the data has been saved

- b. Create Dynamic Scraper (PARTNER 2)

```python
import requests
from bs4 import BeautifulSoup
import pandas as pd
import time
from datetime import datetime

def scrape_enforcement_actions(year, month):
    if year < 2013:
        print("Please set the year to 2013 or later, as only enforcement actions from
```

```
    return

target_date = datetime(year, month, 1)
today = datetime.today()

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

while not stop_scraping:
    url = f"{base_url}?page={page}"
    response = requests.get(url)
    soup = BeautifulSoup(response.text, 'html.parser')

    actions = soup.select("li.usa-card")
    if not actions:
        print("No more data, stopping scraping.")
        break

    for action in actions:
        title = action.select_one("h2.usa-card__heading a").get_text(strip=True)
        link = "https://oig.hhs.gov" + action.select_one("h2.usa-card__heading a")

        date_text = action.select_one("span.text-base-dark.padding-right-105").get
        action_date = datetime.strptime(date_text, "%B %d, %Y")

        if action_date < target_date:
            print(f"Reached data prior to the target date (Target Date: {target_da
            stop_scraping = True
            break

        category = action.select_one("ul.display-inline.add-list-reset li").get_te

        detail_response = requests.get(link)
        detail_soup = BeautifulSoup(detail_response.text, 'html.parser')

        spans = detail_soup.select("span.padding-right-2.text-base")
        agency = spans[1].find_parent("li").get_text(strip=True).replace("Agency:"

        data.append({
            "Title": title,
            "Date": date_text,
            "Category": category,
            "Link": link,
            "Agency": agency
        })

    if stop_scraping:
        break

    time.sleep(1)
    page += 1
```

```
    df = pd.DataFrame(data)

    save_path = f"/Users/suyuanfang/Desktop/Pyhton/ps5/enforcement_actions_{year}_{mon
    df.to_csv(save_path, index=False)
```

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

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

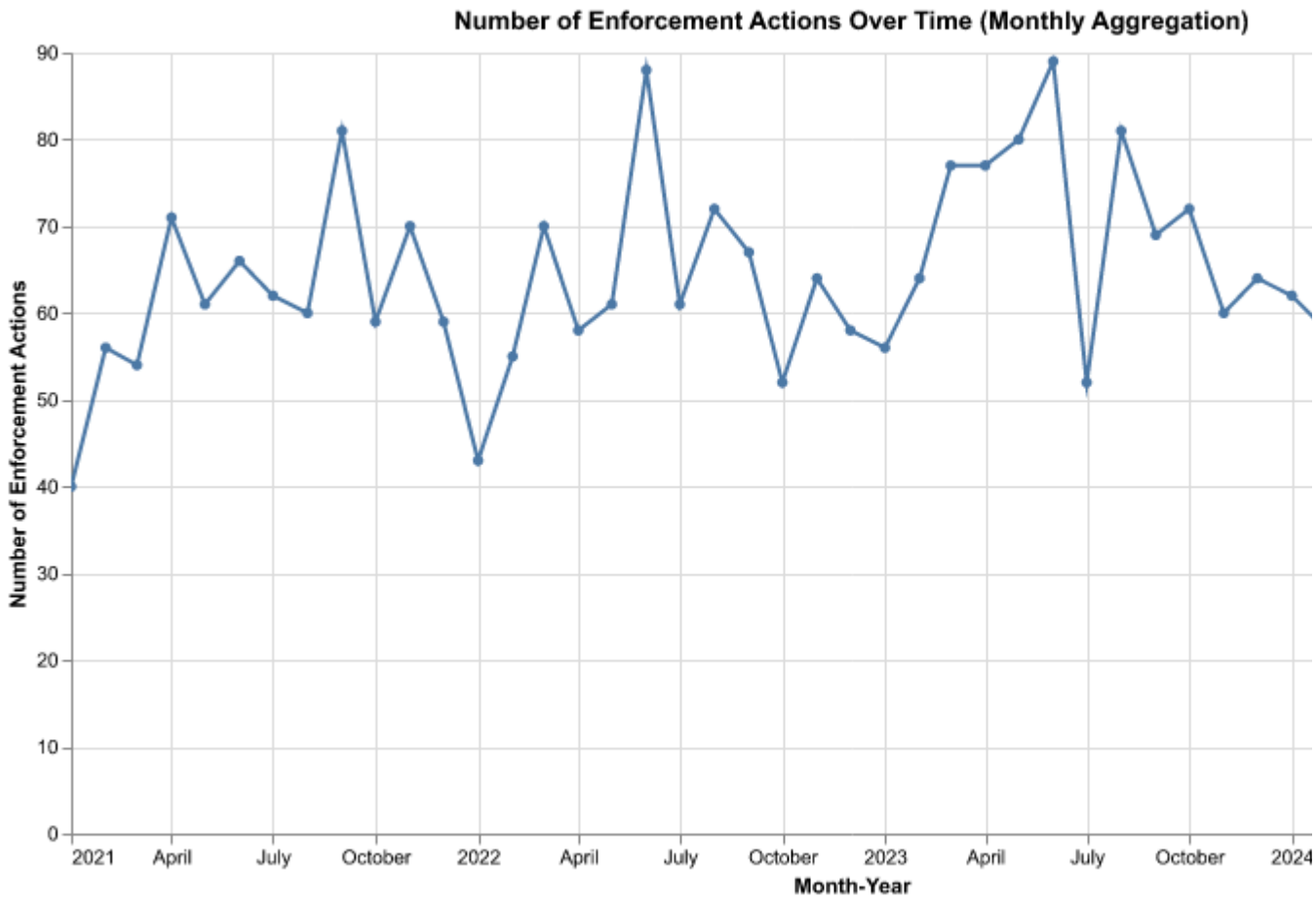
```
df = pd.read_csv('/Users/suyuanfang/Desktop/Pyhton/ps5/enforcement_actions_2021_1.csv'

df['Date'] = pd.to_datetime(df['Date'], format="%B %d, %Y")
num_enforcement_actions = df.shape[0]
earliest_action = df.sort_values("Date").iloc[0]
num_enforcement_actions, earliest_action.to_dict()
```

```
(3022,
 {'Title': 'The United States And Tennessee Resolve Claims With Three Providers For
False Claims Act Liability Relating To ‘P-Stim’ Devices For A Total Of $1.72 Million',
  'Date': Timestamp('2021-01-04 00:00:00'),
  'Category': 'Criminal and Civil Actions',
  'Link': 'https://oig.hhs.gov/fraud/enforcement/the-united-states-and-tennessee-
resolve-claims-with-three-providers-for-false-claims-act-liability-relating-to-p-stim-
devices-for-a-total-of-172-million/',
  'Agency': "U.S. Attorney's Office, Middle District of Tennessee"})
```

Step 3: Plot data based on scraped data

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

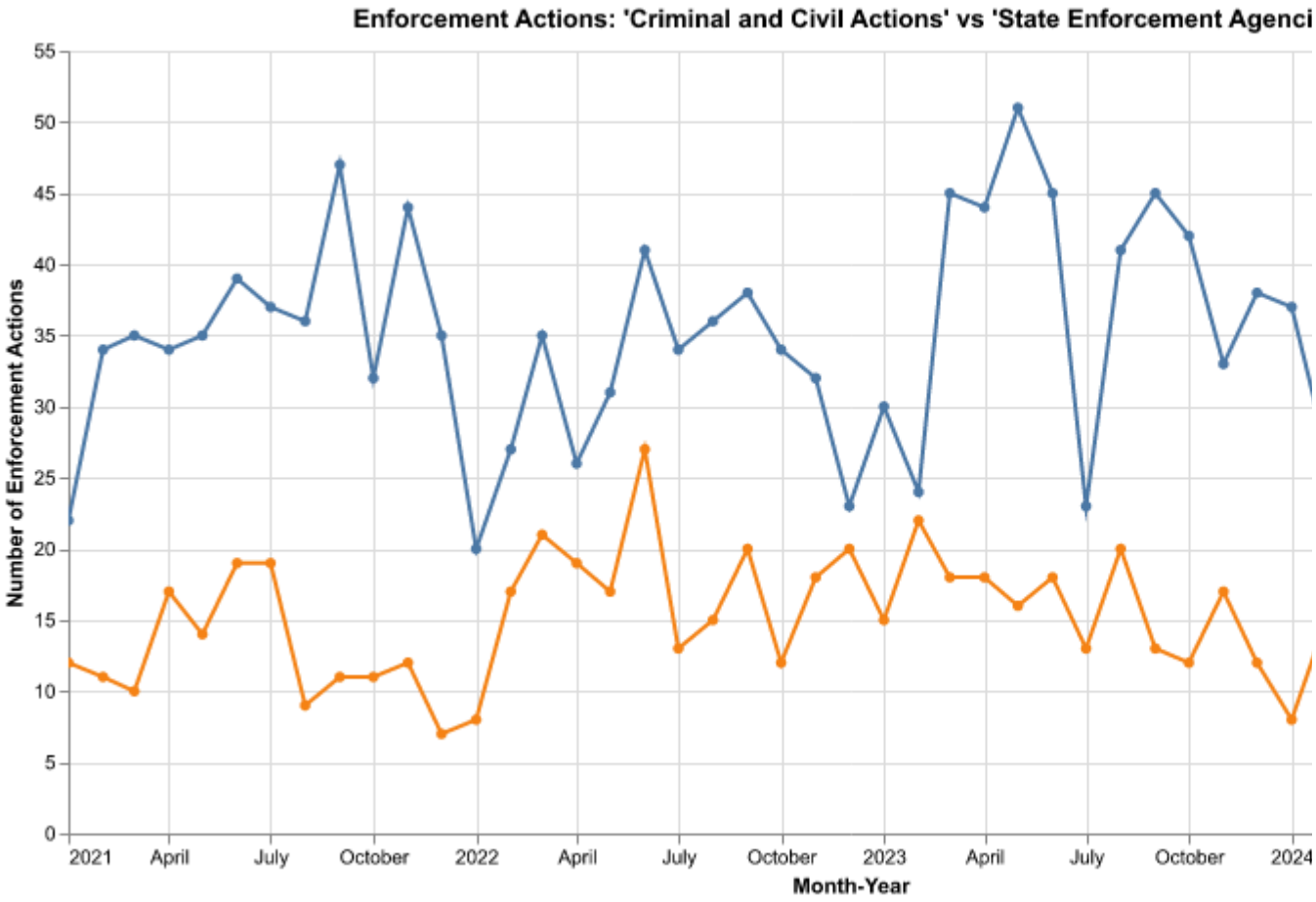
```
import altair as alt
df['YearMonth'] = df['Date'].dt.to_period('M').dt.to_timestamp()
monthly_counts = df.groupby('YearMonth').size().reset_index(name='Counts')
chart = alt.Chart(monthly_counts).mark_line(point=True).encode(
    x=alt.X('YearMonth:T', title='Month-Year'),
    y=alt.Y('Counts', title='Number of Enforcement Actions'),
    tooltip=['YearMonth', 'Counts']
).properties(
    title='Number of Enforcement Actions Over Time (Monthly Aggregation)',
    width=800,
    height=400
).interactive()
chart.show()
```



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

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

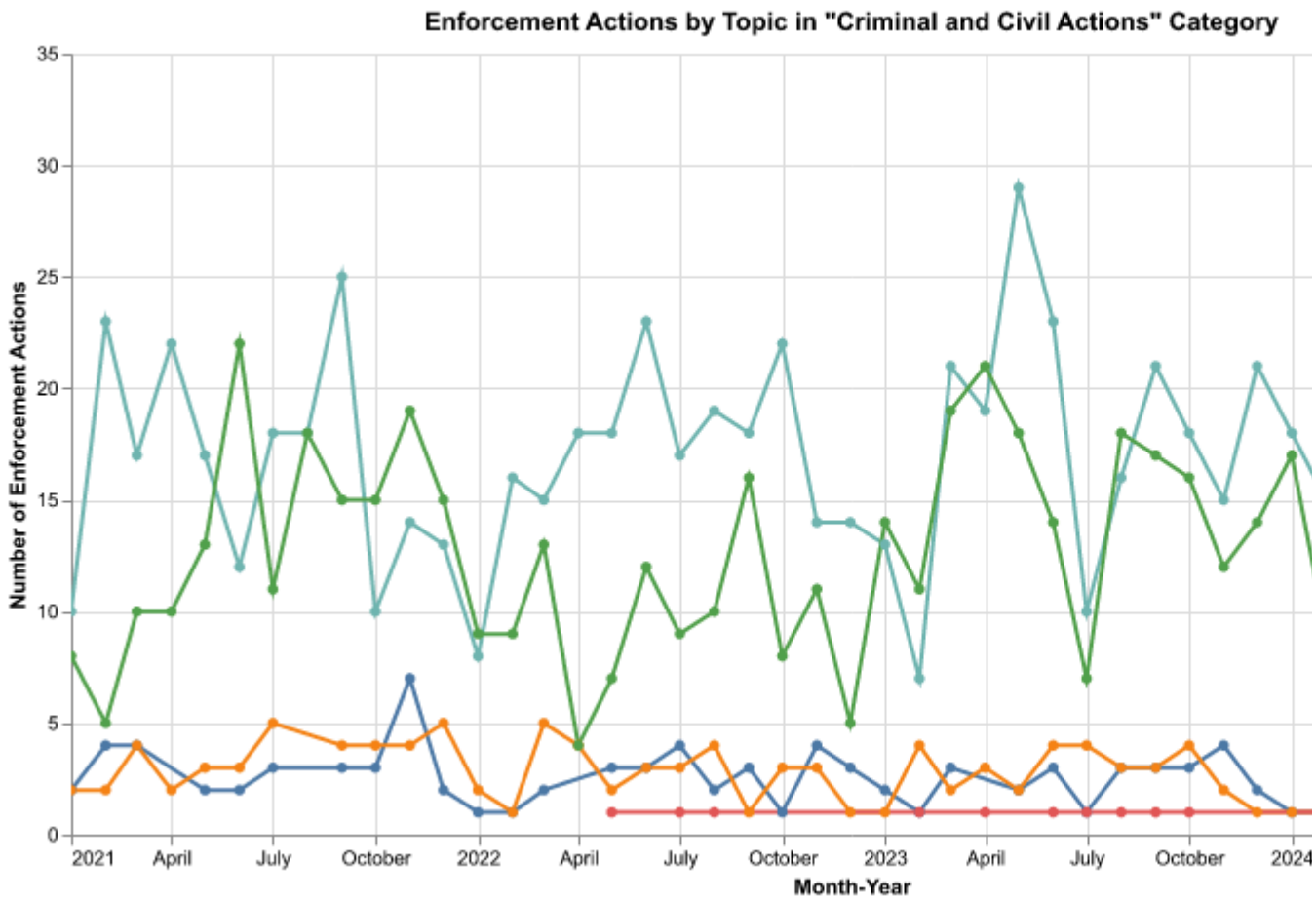
```
filtered_df = df[df['Category'].isin(['Criminal and Civil Actions', 'State Enforcement
filtered_category_monthly_counts = filtered_df.groupby(['YearMonth', 'Category']).size

chart_filtered_category = alt.Chart(filtered_category_monthly_counts).mark_line(point=
    x=alt.X('YearMonth:T', title='Month-Year'),
    y=alt.Y('Counts', title='Number of Enforcement Actions'),
    color='Category:N',
    tooltip=['YearMonth', 'Category', 'Counts']
).properties(
    title="Enforcement Actions: 'Criminal and Civil Actions' vs 'State Enforcement Age
    width=800,
    height=400
).interactive()

chart_filtered_category
```



- based on five topics

```
def categorize_topic(title):
    title = title.lower()
    if "health" in title or "care" in title:
        return "Health Care Fraud"
    elif "financial" in title or "bank" in title or "money" in title:
        return "Financial Fraud"
    elif "drug" in title or "narcotic" in title or "opioid" in title:
        return "Drug Enforcement"
    elif "bribery" in title or "corruption" in title or "kickback" in title:
        return "Bribery/Corruption"
    else:
        return "Other"

df['Topic'] = df.apply(lambda row: categorize_topic(row['Title'])
if row['Category'] == "Criminal and Civil Actions" else None, axis=1)

filtered_topics_df = df[(df['Category'] == "Criminal and Civil Actions") & (df['Topic'

filtered_topics_df['YearMonth'] = filtered_topics_df['Date'].dt.to_period('M').dt.to_t

monthly_topic_counts = filtered_topics_df.groupby(['YearMonth', 'Topic']).size().reset

chart_topic_split = alt.Chart(monthly_topic_counts).mark_line(point=True).encode(
    x=alt.X('YearMonth:T', title='Month-Year'),
    y=alt.Y('Counts', title='Number of Enforcement Actions'),
```

```
    color='Topic:N',
    tooltip=['YearMonth', 'Topic', 'Counts']
).properties(
    title='Enforcement Actions by Topic in "Criminal and Civil Actions" Category',
    width=800,
    height=400
).interactive()

chart_topic_split
```

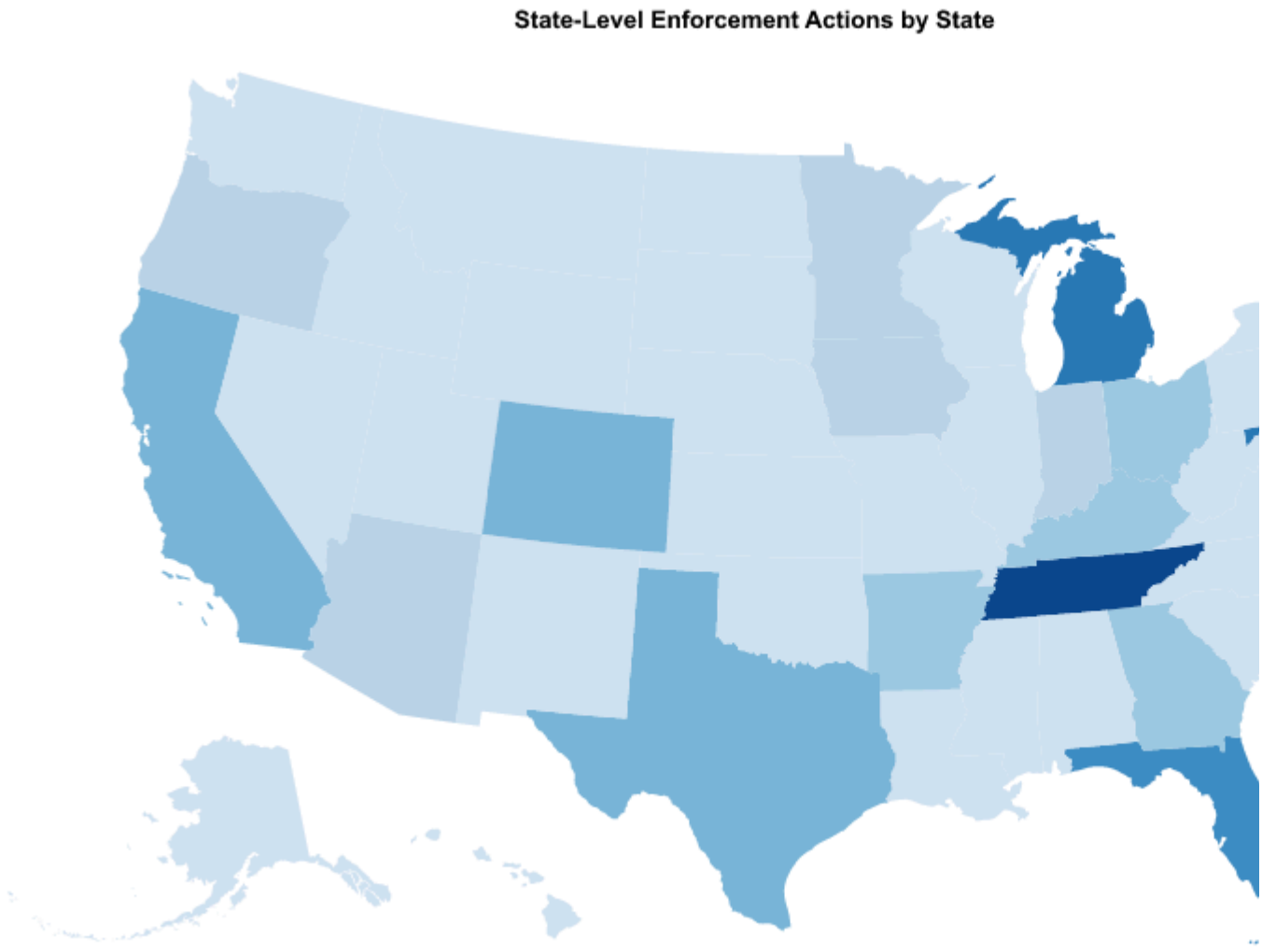


Step 4: Create maps of enforcement activity

1. Map by State (PARTNER 1)

```
import pandas as pd
import geopandas as gpd
import altair as alt
import re
df = pd.read_csv('/Users/suyuanfang/Desktop/Pyhton/ps5/enforcement_actions_2021_1.csv'

state_actions = df[df['Agency'].str.contains("State of", na=False, case=False)]

def extract_state(agency):
```

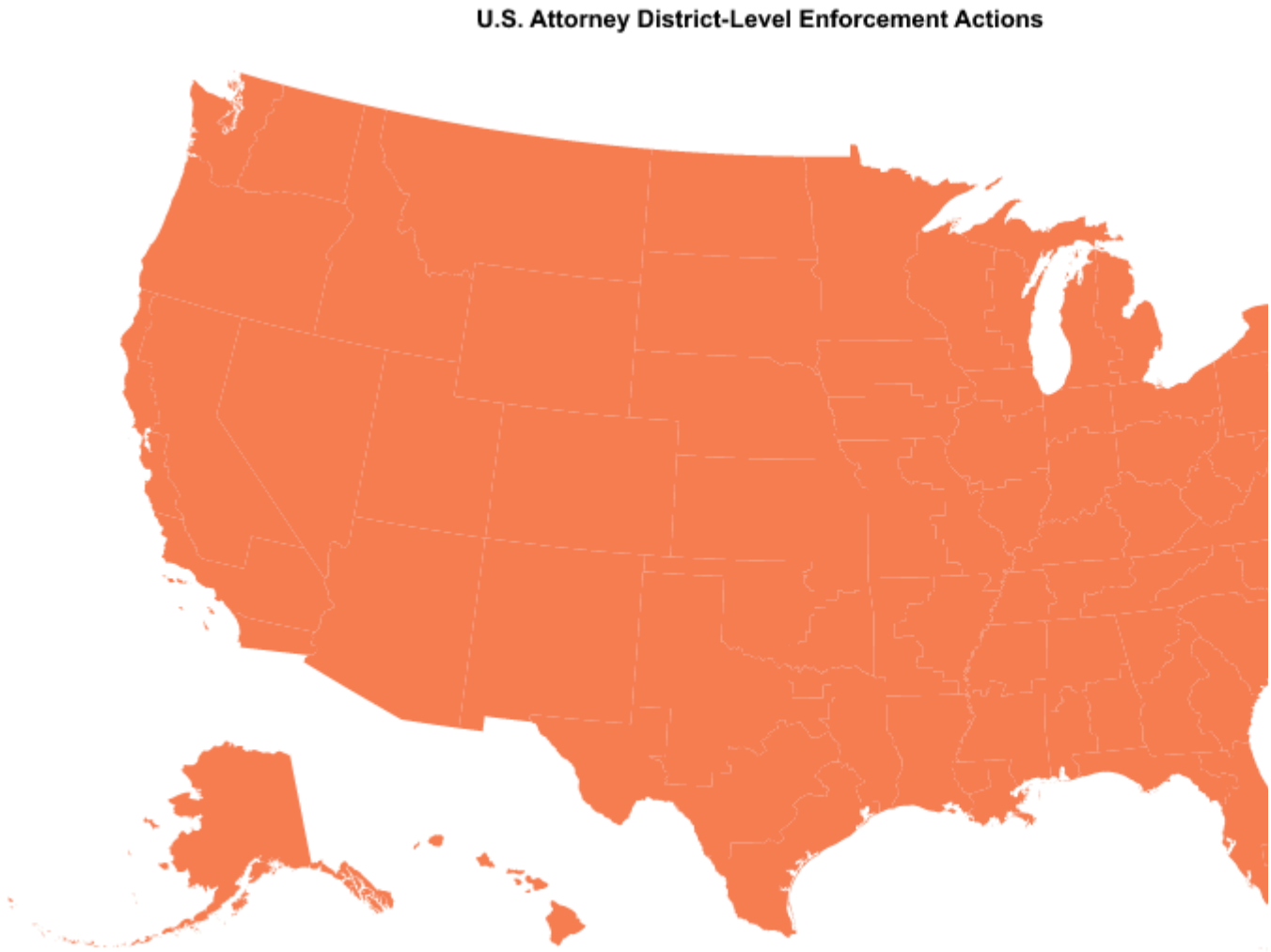
```
    match = re.search(r"State of (\w+)", agency)
    return match.group(1) if match else None

state_actions['State'] = state_actions['Agency'].apply(extract_state)

state_counts = state_actions['State'].value_counts().reset_index()
state_counts.columns = ['State', 'Counts']
states_gdf = gpd.read_file('/Users/suyuanfang/Desktop/Pyhton/ps5/cb_2021_us_state_20m/

states_gdf['STATE_NAME'] = states_gdf['NAME'].str.strip()
merged_gdf = states_gdf.merge(state_counts, left_on='STATE_NAME', right_on='State', ho

geojson_data = merged_gdf.__geo_interface__

chart = alt.Chart(alt.Data(values=geojson_data['features'])).mark_geoshape().encode(
    color=alt.Color('properties.Counts:Q', scale=alt.Scale(scheme='blues'), title="Enf
    tooltip=['properties.STATE_NAME:N', 'properties.Counts:Q']
).properties(
    width=800,
    height=500,
    title="State-Level Enforcement Actions by State"
).project(
    type='albersUsa'
)

chart
```



2. Map by District (PARTNER 2)

```
district_actions = df[df['Agency'].str.contains("District", na=False, case=False)]

def extract_district(agency):
    match = re.search(r"District (?:of|for) (.+)", agency)
    return match.group(1).strip() if match else None

district_actions['District'] = district_actions['Agency'].apply(extract_district)
district_counts = district_actions['District'].value_counts().reset_index()
district_counts.columns = ['District', 'Counts']

districts_gdf = gpd.read_file('/Users/suyuanfang/Desktop/Pyhton/ps5/US Attorney Distri
districts_gdf['district_n'] = districts_gdf['district_n'].str.strip()
merged_gdf = districts_gdf.merge(district_counts, left_on='district_n', right_on='Dist

geojson_data = merged_gdf.__geo_interface__

chart = alt.Chart(alt.Data(values=geojson_data['features'])).mark_geoshape().encode(
    color=alt.Color('properties.Counts:Q', scale=alt.Scale(scheme='orangered'), title=
    tooltip=['properties.DISTRICT_NAME:N', 'properties.Counts:Q']
).properties(
```

```
    width=800,
    height=500,
    title="U.S. Attorney District-Level Enforcement Actions"
).project(
    type='albersUsa'
)

chart
```



Extra Credit

1. Merge zip code shapefile with population

2. Conduct spatial join

3. Map the action ratio in each district