CS109B Data Science 2: Advanced Topics in Data Science

Project D - Predicting Disease Activity

Harvard University
Fall 2020
Instructors: Mark Glickman, Pavlos Protopapas, & Chris Tanner

Group 43
Team: Nisrine Elhauz,

In [1]:

```
#RUN THIS CELL
import requests
from IPython.core.display import HTML
styles = requests.get("https://raw.githubusercontent.com/Harvard-IACS/2018-CS109A/master/content/styles/cs109.css").text
HTML(styles)
```

Out[1]:

In [2]:

```
import pandas as pd
import numpy as np

from datetime import datetime
from sklearn.metrics import f1_score, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA

import matplotlib.pyplot as plt
plt.style.use("ggplot")
```

Reading Covid19 data for cases and deaths for US from nytimes

Reading Covid19 data for cases and deaths for US from nytimes into dataframe.

1. Convert date column to datetime type
2. Validate the date range and the number of days for cases and deaths observations
3. We will use FIPS state code (two-letter alphabetic codes defined in U.S. Federal Information Processing Standard Publication) as the standard for state representation.

In [3]:

```
#Reading Covid19 data for cases and death for US
cases_url ="https://raw.githubusercontent.com/nytimes/covid-19-data/master/us-states.csv"
```

In [4]:

```
#checking date range for Covid 19 data
cases_df = pd.read_csv(cases_url)

#convert date to datetime format
cases_df['date']=pd.to_datetime(cases_df['date'])
cases_df.head()
```

Out[4]:

	date	state	fips	cases	deaths
0	2020-01-21	Washington	53	1	0
1	2020-01-22	Washington	53	1	0
2	2020-01-23	Washington	53	1	0
3	2020-01-24	Illinois	17	1	0
4	2020-01-24	Washington	53	1	0

In [5]:

```
#Getting date range and total number of days
print("Date range for our dataset from", cases_df['date'].min() ,"to", cases_df['date'].max(),
      "with total of days", cases_df['date'].max() - cases_df['date'].min())
```

```
Date range for our dataset from 2020-01-21 00:00:00 to 2020-04-12 00
:00:00 with total of days 82 days 00:00:00
```

In [6]:

```
#Verifying the data: we can notice that fips has max value of 78 which means it
contains
#the data for Outlying area under U.S. sovereignty
cases_df.describe()
```

Out[6]:

	fips	cases	deaths
count	2273.000000	2273.000000	2273.000000
mean	31.109987	2514.911131	75.628685
std	18.186273	11459.131338	452.673761
min	1.000000	0.000000	0.000000
25%	17.000000	11.000000	0.000000
50%	31.000000	144.000000	2.000000
75%	46.000000	1066.000000	21.000000
max	78.000000	188694.000000	9385.000000

Reading Population data from Census US

Reading population data into dataframe.

1. We will use FIPS state code (two-letter alphabetic codes defined in U.S. Federal Information Processing Standard Publication) as the standard for state representation.

In [7]:

```
#Reading population for US
population_url ="https://www2.census.gov/programs-surveys/popest/datasets/2010-2
019/state/detail/SCPRC-EST2019-18+POP-RES.csv"
```

In [8]:

```
population_df = pd.read_csv(population_url)
population_df.rename(columns={'STATE': 'fips'}, inplace = True)
population_df.head()
```

Out[8]:

	SUMLEV	REGION	DIVISION	fips	NAME	POPESTIMATE2019	POPEST18PLUS2019	PCN
0	10	0	0	0	United States	328239523	255200373	
1	40	3	6	1	Alabama	4903185	3814879	
2	40	4	9	2	Alaska	731545	551562	
3	40	4	8	4	Arizona	7278717	5638481	
4	40	3	7	5	Arkansas	3017804	2317649	

In [9]:

```
#Verifying the data: we can notice that fips has max value of 72 which means it
does not contains fips 78
population_df.describe()
```

Out[9]:

	SUMLEV	fips	POPESTIMATE2019	POPEST18PLUS2019	PCNT_POPEST18PLUS
count	53.000000	53.000000	5.300000e+01	5.300000e+01	53.000000
mean	39.433962	29.226415	1.244666e+07	9.679655e+06	77.937736
std	4.120817	17.108974	4.479917e+07	3.482593e+07	2.043021
min	10.000000	0.000000	5.787590e+05	4.450250e+05	71.000000
25%	40.000000	16.000000	1.792147e+06	1.432580e+06	76.800000
50%	40.000000	29.000000	4.467673e+06	3.464802e+06	77.900000
75%	40.000000	42.000000	7.614893e+06	5.951832e+06	79.100000
max	40.000000	72.000000	3.282395e+08	2.552004e+08	82.100000

In [10]:

```
df = pd.merge(cases_df,population_df[['fips','POPESTIMATE2019','POPEST18PLUS2019
','PCNT_POPEST18PLUS']],on='fips',how='right',indicator=True)
df.head()
```

Out[10]:

	date	state	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_PO
0	2020-01-21	Washington	53	1.0	0.0	7614893	5951832	
1	2020-01-22	Washington	53	1.0	0.0	7614893	5951832	
2	2020-01-23	Washington	53	1.0	0.0	7614893	5951832	
3	2020-01-24	Washington	53	1.0	0.0	7614893	5951832	
4	2020-01-25	Washington	53	1.0	0.0	7614893	5951832	

In [11]:

```
#grouping data by date and getting the total number of cases and deaths
total_df = cases_df.groupby('date').sum()#.reset_index()
total_df.head()
```

Out[11]:

date	fips	cases	deaths
2020-01-21	53	1	0
2020-01-22	53	1	0
2020-01-23	53	1	0
2020-01-24	70	2	0
2020-01-25	76	3	0

In [12]:

```
total_df.describe()
```

Out[12]:

	fips	cases	deaths
count	83.000000	83.000000	83.000000
mean	851.963855	68872.204819	2071.132530
std	731.508889	141965.944901	4950.433173
min	53.000000	1.000000	0.000000
25%	160.000000	13.000000	0.000000
50%	467.000000	104.000000	6.000000
75%	1693.000000	38280.000000	491.000000
max	1822.000000	555371.000000	22056.000000

Merging Covid 19 date with population data

1. US population record will be dropped
2. validate available fips in both dataframe
3. After data merging validation, we will drop _merge column

In [13]:

```
#check record with fips that exist in cases and not in population
result_df=cases_df.fips.isin(population_df.fips).astype(int)
print (result_df.value_counts())
```

```
1    2194
0      79
Name: fips, dtype: int64
```

In [14]:

```
df = pd.merge(cases_df,population_df[['fips','POPESTIMATE2019','POPEST18PLUS2019
','PCNT_POPEST18PLUS']],on='fips',how='right',indicator=True)
df.head()
```

Out[14]:

	date	state	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_PO
0	2020-01-21	Washington	53	1.0	0.0	7614893	5951832	
1	2020-01-22	Washington	53	1.0	0.0	7614893	5951832	
2	2020-01-23	Washington	53	1.0	0.0	7614893	5951832	
3	2020-01-24	Washington	53	1.0	0.0	7614893	5951832	
4	2020-01-25	Washington	53	1.0	0.0	7614893	5951832	

In [15]:

```
#drop invalid record & merge column
df = df.dropna()
df.drop(['_merge'], axis=1) #2217 2142
```

Out[15]:

	date	state	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_
0	2020-01-21	Washington	53	1.0	0.0	7614893	5951832	
1	2020-01-22	Washington	53	1.0	0.0	7614893	5951832	
2	2020-01-23	Washington	53	1.0	0.0	7614893	5951832	
3	2020-01-24	Washington	53	1.0	0.0	7614893	5951832	
4	2020-01-25	Washington	53	1.0	0.0	7614893	5951832	
...	...	...	...	...	...	...	...	
2189	2020-04-08	West Virginia	54	483.0	4.0	1792147	1432580	
2190	2020-04-09	West Virginia	54	524.0	5.0	1792147	1432580	
2191	2020-04-10	West Virginia	54	537.0	5.0	1792147	1432580	
2192	2020-04-11	West Virginia	54	593.0	6.0	1792147	1432580	
2193	2020-04-12	West Virginia	54	615.0	8.0	1792147	1432580	

2194 rows × 8 columns

Calculating log10 for cases & calculating cases , deaths per 100k

In [16]:

```
df['cases_log10'] = np.log10(df['cases'])
df['cases_log10'].loc[np.isinf(df['cases_log10'])] = 0
df['cases_per_100k'] = df['cases']/df['POPESTIMATE2019']*1e5
df['deaths_per_100k'] = df['deaths']/df['POPESTIMATE2019']*1e5
df.head()
df.sort_values(by=['fips'])
```

```
/anaconda3/envs/cs109b/lib/python3.7/site-packages/pandas/core/index
ing.py:670: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pand
as-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
py
  self._setitem_with_indexer(indexer, value)
```

Out[16]:

	date	state	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_P
2069	2020-04-08	Alabama	1	2499.0	67.0	4903185	3814879	
2055	2020-03-25	Alabama	1	386.0	1.0	4903185	3814879	
2056	2020-03-26	Alabama	1	538.0	3.0	4903185	3814879	
2057	2020-03-27	Alabama	1	639.0	4.0	4903185	3814879	
2058	2020-03-28	Alabama	1	720.0	4.0	4903185	3814879	
...	...	...	...	...	...	...	...	
2151	2020-03-28	Puerto Rico	72	100.0	3.0	3193694	2620963	
2150	2020-03-27	Puerto Rico	72	79.0	3.0	3193694	2620963	
2149	2020-03-26	Puerto Rico	72	64.0	2.0	3193694	2620963	
2165	2020-04-11	Puerto Rico	72	788.0	42.0	3193694	2620963	
2145	2020-03-22	Puerto Rico	72	23.0	1.0	3193694	2620963	

2194 rows × 12 columns

Visualizing data

1. Visualizing total cases & deaths per date
2. Visualizing US map with total cases
3. calculating the cases & deathes per 100k population and log10

In [17]:

```
total_df = df.groupby('date').sum().reset_index()
total_df.head()
```

Out[17]:

	date	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_POPEST18PLUS
0	2020-01-21	53	1.0	0.0	7614893	5951832	78.2
1	2020-01-22	53	1.0	0.0	7614893	5951832	78.2
2	2020-01-23	53	1.0	0.0	7614893	5951832	78.2
3	2020-01-24	70	2.0	0.0	20286714	15805778	156.0
4	2020-01-25	76	3.0	0.0	59798937	46423360	233.5

In [18]:

```
total_df.sort_values(by=['fips'])
```

Out[18]:

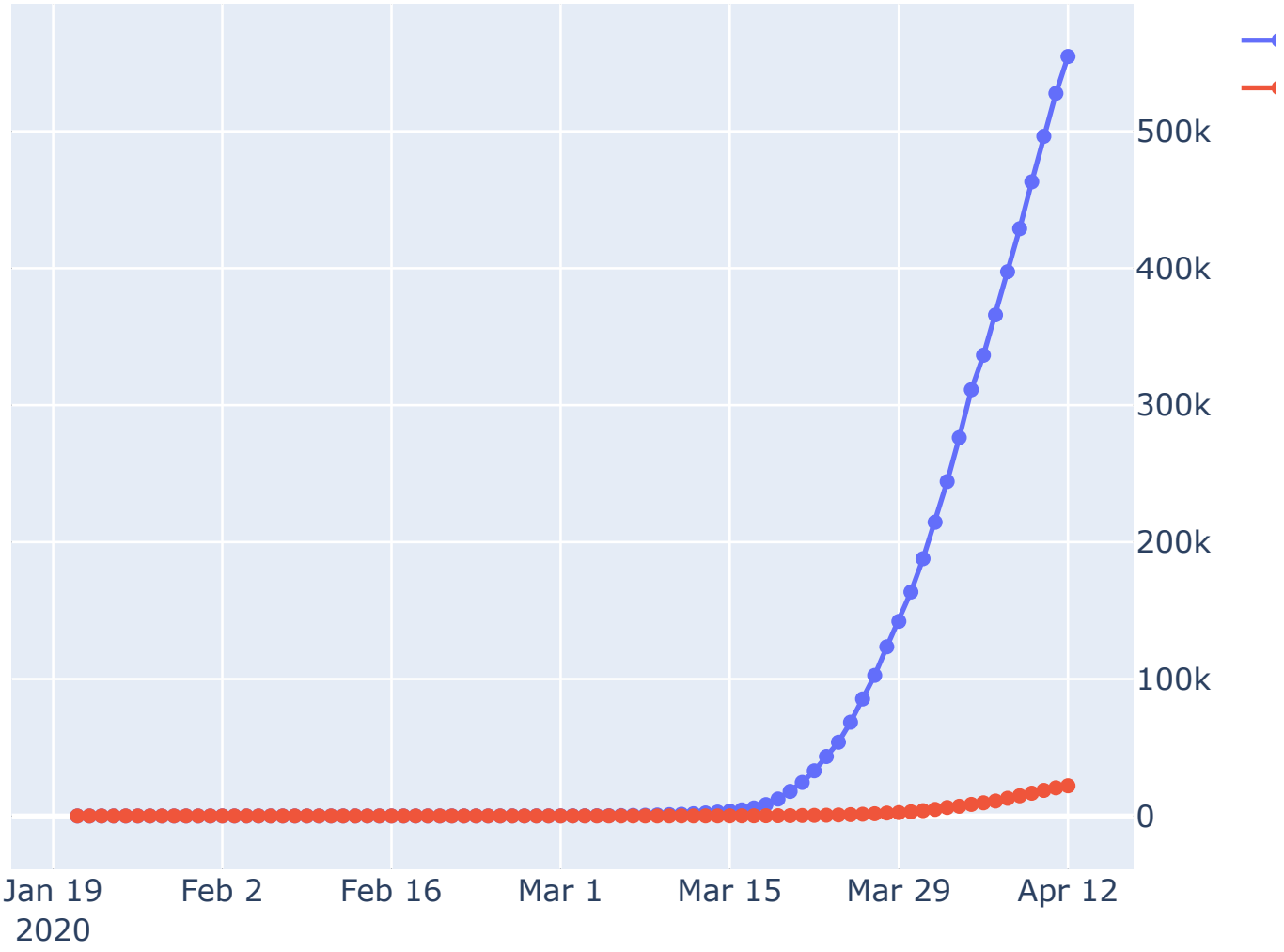
	date	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_POPEST18
0	2020-01-21	53	1.0	0.0	7614893	5951832	
1	2020-01-22	53	1.0	0.0	7614893	5951832	
2	2020-01-23	53	1.0	0.0	7614893	5951832	
3	2020-01-24	70	2.0	0.0	20286714	15805778	
4	2020-01-25	76	3.0	0.0	59798937	46423360	
...	...	...	...	...	...	...	
58	2020-03-19	1549	12398.0	203.0	331433217	257821336	4
57	2020-03-18	1549	8334.0	157.0	331433217	257821336	4
56	2020-03-17	1549	5900.0	116.0	331433217	257821336	4
68	2020-03-29	1549	142111.0	2485.0	331433217	257821336	4
82	2020-04-12	1549	554593.0	22048.0	331433217	257821336	4

83 rows × 10 columns

In [19]:

```
from plotly.subplots import make_subplots
import plotly.graph_objects as go

fig = make_subplots(specs=[[{"secondary_y": True}]])

fig.add_trace(
    go.Scatter(x=total_df['date'], y=total_df['cases'],mode='lines+markers', nam
e='Total Cases'),
    secondary_y=True
)

fig.add_trace(
    go.Scatter(x=total_df['date'], y=total_df['deaths'],mode='lines+markers',nam
e='Total Deaths'),
    secondary_y=True
)

fig.update_layout(
    title={
        'text': "Total Confirmed COVID-19 Cases in the US",
        'y':0.9,
        'x':0.5,
        'xanchor': 'center',
        'yanchor': 'top'})

fig.show()
```

Total Confirmed COVID-19 Cases in the US



In [20]:

```
df_state = df.groupby(
   ['fips']
).agg(
    {
         'fips':'first',
         'cases':sum,    # Sum duration per group
         'deaths': sum,  # get the count of networks
         'POPESTIMATE2019': 'first',  # get the first date per group
         'POPEST18PLUS2019':'first',
         'PCNT_POPEST18PLUS':'first',
          'cases_log10':'first',
         'cases_per_100k':'first',
         'deaths_per_100k':'first',
    }
)

df_state.head()
```

Out[20]:

	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_POPEST18PLUS
fips						
1	1	32907.0	756.0	4903185	3814879	77.8
2	2	3275.0	55.0	731545	551562	75.4
4	4	36762.0	979.0	7278717	5638481	77.5
5	5	15467.0	249.0	3017804	2317649	76.8
6	6	258262.0	6306.0	39512223	30617582	77.5

In [21]:

```
df_state.describe()
```

Out[21]:

	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT
count	52.000000	5.200000e+01	52.000000	5.200000e+01	5.200000e+01	
mean	29.788462	1.098179e+05	3304.230769	6.373716e+06	4.958103e+06	
std	16.774557	3.000033e+05	10101.760450	7.301997e+06	5.650415e+06	
min	1.000000	3.275000e+03	0.000000	5.787590e+05	4.450250e+05	
25%	16.750000	1.243675e+04	244.750000	1.790876e+06	1.409151e+06	
50%	29.500000	3.005850e+04	765.000000	4.342705e+06	3.407988e+06	
75%	42.500000	8.820725e+04	2463.000000	7.362761e+06	5.716819e+06	
max	72.000000	2.100768e+06	71311.000000	3.951222e+07	3.061758e+07	

In [22]:

```
total_df.sort_values(by=['fips'])
```

Out[22]:

	date	fips	cases	deaths	POPESTIMATE2019	POPEST18PLUS2019	PCNT_POPEST18
0	2020-01-21	53	1.0	0.0	7614893	5951832	
1	2020-01-22	53	1.0	0.0	7614893	5951832	
2	2020-01-23	53	1.0	0.0	7614893	5951832	
3	2020-01-24	70	2.0	0.0	20286714	15805778	
4	2020-01-25	76	3.0	0.0	59798937	46423360	
...	...	...	...	...	...	...	
58	2020-03-19	1549	12398.0	203.0	331433217	257821336	4
57	2020-03-18	1549	8334.0	157.0	331433217	257821336	4
56	2020-03-17	1549	5900.0	116.0	331433217	257821336	4
68	2020-03-29	1549	142111.0	2485.0	331433217	257821336	4
82	2020-04-12	1549	554593.0	22048.0	331433217	257821336	4

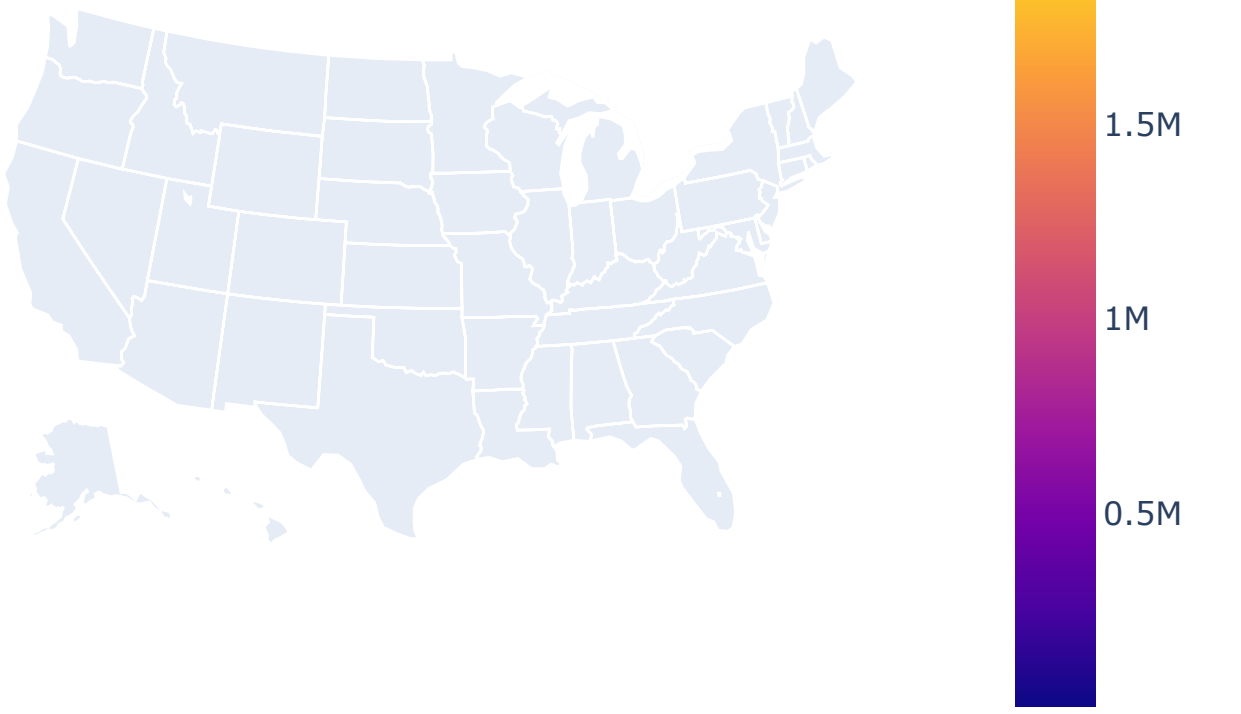
83 rows × 10 columns

In [23]:

```
#https://plotly.com/python/mapbox-county-choropleth/
import plotly

fig = go.Figure(data=go.Choropleth(
    locations=df_state['fips'],
    z=df_state['cases'],
    locationmode='USA-states',
    colorscale='Reds',
    autocolorscale=True,
    text=df['cases_per_100k'],
    marker_line_color='white', # line markers between states
    colorbar_title="COVID-19 cases per 100,000"
))

fig.update_layout(
    title_text='Number of COVID-19 cases per 100,000 people by state',
    geo = dict(
        scope='usa',
        projection=go.layout.geo.Projection(type = 'albers usa'),
        showlakes=True, # lakes
        lakecolor='rgb(255, 255, 255)'),
)

#plotly.offline.plot(fig)
```

```
fig.show()
```

Number of COVID-19 cases per 100,000 people by state



In [24]:

```
from urllib.request import urlopen
import json
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json') as response:
    counties = json.load(response)


import plotly.express as px

fig = px.choropleth_mapbox(df_state, geojson=counties, locations='fips', color='cases',
                           color_continuous_scale="Viridis",
                           range_color=(0, 12),
                           mapbox_style="carto-positron",
                           zoom=3, center = {"lat": 37.0902, "lon": -95.7129},
                           opacity=0.5,
                           labels={''}
                          )
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

In [ ]: