CS6220_Final_Project

August 5, 2020

[1]:
```
# imports
import numpy as np
import csv
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import date, timedelta, datetime
import scipy.stats as stats
from matplotlib import cm
import random
from scipy.stats import zscore
from sklearn import preprocessing
import math
from matplotlib import style
import matplotlib as mpl
```

1 Mobility/Restaurant Data (International)

[2]:
```
# First, create a Dataframe using the Apple mobility data
apple_mob_file = "data/mobility-trends.csv"
apple_df = pd.read_csv(apple_mob_file)

apple_df.head()
```

[2]:

	geo_type	region	transportation_type	date	value
0	country/region	Albania	driving	2020-01-13	100.00
1	country/region	Albania	driving	2020-01-14	95.30
2	country/region	Albania	driving	2020-01-15	101.43
3	country/region	Albania	driving	2020-01-16	97.20
4	country/region	Albania	driving	2020-01-17	103.55

[3]:
```
# Next, create a Dataframe using the Google mobility data. We can combine this␣
 ↪with the Apple Dataframe later.
google_mob_file = "data/regional-mobility.csv"
google_df = pd.read_csv(google_mob_file)
```

```
google_df.head()
```

[3]:
```
                country region        date  retail  grocery_and_pharmacy  \
0  United Arab Emirates  Total  2020-02-15     0.0                   4.0
1  United Arab Emirates  Total  2020-02-16     1.0                   4.0
2  United Arab Emirates  Total  2020-02-17    -1.0                   1.0
3  United Arab Emirates  Total  2020-02-18    -2.0                   1.0
4  United Arab Emirates  Total  2020-02-19    -2.0                   0.0

   parks  transit_stations  workplaces  residential
0    5.0               0.0         2.0          1.0
1    4.0               1.0         2.0          1.0
2    5.0               1.0         2.0          1.0
3    5.0               0.0         2.0          1.0
4    4.0              -1.0         2.0          1.0
```

[4]:
```
# Now, create a Dataframe from the restaurant data.
restaurant_file = "data/restaurant-performance-country.csv"
restaurant_df = pd.read_csv(restaurant_file, index_col=0)

restaurant_df.head()
```

[4]:
```
             country  percent_yoy_change        date
region_type
countries     Global                  -1  2020-02-18
countries     Global                   3  2020-02-19
countries     Global                  -1  2020-02-20
countries     Global                  -2  2020-02-21
countries     Global                   1  2020-02-22
```

[5]:
```
# Finally, let's create a Dataframe of COVID-19 spread data for comparison.

covid_file = "WHO-COVID-19-global-data.csv"
covid_df = pd.read_csv(covid_file)

covid_df.head()
```

[5]:
```
  Date_reported  Country_code      Country  WHO_region   New_cases  \
0    2020-02-24            AF  Afghanistan        EMRO           1
1    2020-02-25            AF  Afghanistan        EMRO           0
2    2020-02-26            AF  Afghanistan        EMRO           0
3    2020-02-27            AF  Afghanistan        EMRO           0
4    2020-02-28            AF  Afghanistan        EMRO           0

    Cumulative_cases   New_deaths   Cumulative_deaths
0                  1            0                   0
```

```
1                 1           0                 0
2                 1           0                 0
3                 1           0                 0
4                 1           0                 0
```

[6]:
```
# Let's combine the Apple and Google data into a single Dataframe
# First, let's sort the instances by country and remove instances without a␣
 ↪country or that do not occur on a national level

apple_df = apple_df[apple_df.geo_type == 'country/region'] # Only keeps␣
 ↪instances of countries, excludes cities
apple_df = apple_df.drop(columns=["geo_type"]) # Delete unnecessary column
apple_df = apple_df.rename(columns={"region": "country"}) # rename column for␣
 ↪accuracy
# print(len(apple_df))

google_df = google_df[google_df.region == 'Total'] # Only keeps data for entire␣
 ↪countries (not regions/cities)
google_df = google_df.drop(columns=["region"]) # Delete unnecessary column
# print(len(google_df))

# Now we can combine the two based on country value and date
mobility_df = pd.merge(apple_df, google_df, how="outer", on=["country", "date"])
mobility_df
# print(len(mobility_df))
```

[6]:
```
        country transportation_type        date   value  retail  \
0       Albania              driving  2020-01-13  100.00     NaN
1       Albania              walking  2020-01-13  100.00     NaN
2       Albania              driving  2020-01-14   95.30     NaN
3       Albania              walking  2020-01-14  100.68     NaN
4       Albania              driving  2020-01-15  101.43     NaN
5       Albania              walking  2020-01-15   98.93     NaN
6       Albania              driving  2020-01-16   97.20     NaN
7       Albania              walking  2020-01-16   98.46     NaN
8       Albania              driving  2020-01-17  103.55     NaN
9       Albania              walking  2020-01-17  100.85     NaN
10      Albania              driving  2020-01-18  112.67     NaN
11      Albania              walking  2020-01-18  100.13     NaN
12      Albania              driving  2020-01-19  104.83     NaN
13      Albania              walking  2020-01-19   82.13     NaN
14      Albania              driving  2020-01-20   94.39     NaN
15      Albania              walking  2020-01-20   95.65     NaN
16      Albania              driving  2020-01-21   94.07     NaN
17      Albania              walking  2020-01-21   97.78     NaN
18      Albania              driving  2020-01-22   93.51     NaN
19      Albania              walking  2020-01-22   95.39     NaN
```

```
20      Albania             driving  2020-01-23   92.94     NaN
21      Albania             walking  2020-01-23   94.24     NaN
22      Albania             driving  2020-01-24  102.13     NaN
23      Albania             walking  2020-01-24   93.73     NaN
24      Albania             driving  2020-01-25  102.38     NaN
25      Albania             walking  2020-01-25   97.06     NaN
26      Albania             driving  2020-01-26  101.41     NaN
27      Albania             walking  2020-01-26   77.27     NaN
28      Albania             driving  2020-01-27   94.62     NaN
29      Albania             walking  2020-01-27   83.37     NaN
...         ...                 ...         ...     ...      ...
20943  Zimbabwe                 NaN  2020-03-19     NaN      0.0
20944  Zimbabwe                 NaN  2020-03-20     NaN     -2.0
20945  Zimbabwe                 NaN  2020-03-21     NaN     -5.0
20946  Zimbabwe                 NaN  2020-03-22     NaN     -8.0
20947  Zimbabwe                 NaN  2020-03-23     NaN     -5.0
20948  Zimbabwe                 NaN  2020-03-24     NaN     -7.0
20949  Zimbabwe                 NaN  2020-03-25     NaN    -18.0
20950  Zimbabwe                 NaN  2020-03-26     NaN    -21.0
20951  Zimbabwe                 NaN  2020-03-27     NaN    -28.0
20952  Zimbabwe                 NaN  2020-03-28     NaN     -9.0
20953  Zimbabwe                 NaN  2020-03-29     NaN     -2.0
20954  Zimbabwe                 NaN  2020-03-30     NaN    -75.0
20955  Zimbabwe                 NaN  2020-03-31     NaN    -71.0
20956  Zimbabwe                 NaN  2020-04-01     NaN    -68.0
20957  Zimbabwe                 NaN  2020-04-02     NaN    -69.0
20958  Zimbabwe                 NaN  2020-04-03     NaN    -70.0
20959  Zimbabwe                 NaN  2020-04-04     NaN    -69.0
20960  Zimbabwe                 NaN  2020-04-05     NaN    -63.0
20961  Zimbabwe                 NaN  2020-04-06     NaN    -66.0
20962  Zimbabwe                 NaN  2020-04-07     NaN    -64.0
20963  Zimbabwe                 NaN  2020-04-08     NaN    -62.0
20964  Zimbabwe                 NaN  2020-04-09     NaN    -60.0
20965  Zimbabwe                 NaN  2020-04-10     NaN    -70.0
20966  Zimbabwe                 NaN  2020-04-11     NaN    -69.0
20967  Zimbabwe                 NaN  2020-04-12     NaN    -64.0
20968  Zimbabwe                 NaN  2020-04-13     NaN    -71.0
20969  Zimbabwe                 NaN  2020-04-14     NaN    -63.0
20970  Zimbabwe                 NaN  2020-04-15     NaN    -62.0
20971  Zimbabwe                 NaN  2020-04-16     NaN    -62.0
20972  Zimbabwe                 NaN  2020-04-17     NaN    -62.0

       grocery_and_pharmacy  parks  transit_stations  workplaces  residential
0                       NaN    NaN               NaN         NaN          NaN
1                       NaN    NaN               NaN         NaN          NaN
2                       NaN    NaN               NaN         NaN          NaN
3                       NaN    NaN               NaN         NaN          NaN
```

4	NaN	NaN	NaN	NaN	NaN
5	NaN	NaN	NaN	NaN	NaN
6	NaN	NaN	NaN	NaN	NaN
7	NaN	NaN	NaN	NaN	NaN
8	NaN	NaN	NaN	NaN	NaN
9	NaN	NaN	NaN	NaN	NaN
10	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN
13	NaN	NaN	NaN	NaN	NaN
14	NaN	NaN	NaN	NaN	NaN
15	NaN	NaN	NaN	NaN	NaN
16	NaN	NaN	NaN	NaN	NaN
17	NaN	NaN	NaN	NaN	NaN
18	NaN	NaN	NaN	NaN	NaN
19	NaN	NaN	NaN	NaN	NaN
20	NaN	NaN	NaN	NaN	NaN
21	NaN	NaN	NaN	NaN	NaN
22	NaN	NaN	NaN	NaN	NaN
23	NaN	NaN	NaN	NaN	NaN
24	NaN	NaN	NaN	NaN	NaN
25	NaN	NaN	NaN	NaN	NaN
26	NaN	NaN	NaN	NaN	NaN
27	NaN	NaN	NaN	NaN	NaN
28	NaN	NaN	NaN	NaN	NaN
29	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...
20943	9.0	-9.0	-11.0	8.0	2.0
20944	8.0	-11.0	-20.0	11.0	1.0
20945	8.0	-6.0	-16.0	7.0	4.0
20946	3.0	-11.0	-21.0	6.0	7.0
20947	10.0	-9.0	-17.0	5.0	6.0
20948	14.0	-13.0	-17.0	-10.0	9.0
20949	3.0	-16.0	-25.0	-20.0	15.0
20950	1.0	-19.0	-34.0	-27.0	18.0
20951	-6.0	-32.0	-44.0	-30.0	23.0
20952	18.0	-14.0	-19.0	-2.0	13.0
20953	24.0	-4.0	-6.0	14.0	13.0
20954	-68.0	-51.0	-80.0	-72.0	46.0
20955	-61.0	-51.0	-82.0	-72.0	44.0
20956	-57.0	-47.0	-80.0	-70.0	44.0
20957	-58.0	-53.0	-80.0	-71.0	45.0
20958	-58.0	-55.0	-81.0	-68.0	47.0
20959	-58.0	-52.0	-80.0	-56.0	37.0
20960	-53.0	-47.0	-77.0	-36.0	32.0
20961	-55.0	-48.0	-79.0	-67.0	44.0
20962	-52.0	-49.0	-78.0	-67.0	42.0

```
20963                -51.0  -51.0             -78.0       -66.0         41.0
20964                -47.0  -47.0             -77.0       -66.0         41.0
20965                -57.0  -53.0             -82.0       -78.0         48.0
20966                -56.0  -48.0             -81.0       -58.0         36.0
20967                -56.0  -45.0             -80.0       -35.0         33.0
20968                -63.0  -51.0             -83.0       -80.0         49.0
20969                -53.0  -47.0             -78.0       -64.0         41.0
20970                -50.0  -51.0             -77.0       -65.0         42.0
20971                -50.0  -48.0             -77.0       -64.0         42.0
20972                -48.0  -47.0             -79.0       -62.0         43.0

[20973 rows x 10 columns]
```

[7]:
```
# Now, let's remove any rows with "null" values (this means they did not appear␣
 ↪in both Dataframes)
mobility_df.dropna()
```

[7]:

	country	transportation_type	date	value	retail	\
276	Argentina	driving	2020-02-15	127.62	1.0	
277	Argentina	walking	2020-02-15	103.47	1.0	
278	Argentina	driving	2020-02-16	88.20	-6.0	
279	Argentina	walking	2020-02-16	69.12	-6.0	
280	Argentina	driving	2020-02-17	92.28	-10.0	
281	Argentina	walking	2020-02-17	93.42	-10.0	
282	Argentina	driving	2020-02-18	98.15	3.0	
283	Argentina	walking	2020-02-18	111.49	3.0	
284	Argentina	driving	2020-02-19	98.96	1.0	
285	Argentina	walking	2020-02-19	107.02	1.0	
286	Argentina	driving	2020-02-20	104.71	0.0	
287	Argentina	walking	2020-02-20	111.36	0.0	
288	Argentina	driving	2020-02-21	132.57	3.0	
289	Argentina	walking	2020-02-21	124.13	3.0	
290	Argentina	driving	2020-02-22	141.76	6.0	
291	Argentina	walking	2020-02-22	120.44	6.0	
292	Argentina	driving	2020-02-23	108.21	14.0	
293	Argentina	walking	2020-02-23	100.50	14.0	
294	Argentina	driving	2020-02-24	107.92	-10.0	
295	Argentina	walking	2020-02-24	96.60	-10.0	
296	Argentina	driving	2020-02-25	99.51	-20.0	
297	Argentina	walking	2020-02-25	82.59	-20.0	
298	Argentina	driving	2020-02-26	96.38	1.0	
299	Argentina	walking	2020-02-26	107.79	1.0	
300	Argentina	driving	2020-02-27	97.96	-1.0	
301	Argentina	walking	2020-02-27	110.62	-1.0	
302	Argentina	driving	2020-02-28	116.77	3.0	
303	Argentina	walking	2020-02-28	124.18	3.0	
304	Argentina	driving	2020-02-29	117.72	5.0	

305	Argentina	walking	2020-02-29	108.49	5.0
...	...	...	...	...	...
16017	Vietnam	driving	2020-04-03	41.82	-63.0
16018	Vietnam	walking	2020-04-03	43.79	-63.0
16019	Vietnam	driving	2020-04-04	41.01	-65.0
16020	Vietnam	walking	2020-04-04	43.99	-65.0
16021	Vietnam	driving	2020-04-05	42.45	-67.0
16022	Vietnam	walking	2020-04-05	44.22	-67.0
16023	Vietnam	driving	2020-04-06	44.08	-60.0
16024	Vietnam	walking	2020-04-06	45.12	-60.0
16025	Vietnam	driving	2020-04-07	44.90	-61.0
16026	Vietnam	walking	2020-04-07	45.84	-61.0
16027	Vietnam	driving	2020-04-08	44.87	-60.0
16028	Vietnam	walking	2020-04-08	45.94	-60.0
16029	Vietnam	driving	2020-04-09	45.84	-61.0
16030	Vietnam	walking	2020-04-09	46.74	-61.0
16031	Vietnam	driving	2020-04-10	47.82	-58.0
16032	Vietnam	walking	2020-04-10	48.89	-58.0
16033	Vietnam	driving	2020-04-11	46.42	-59.0
16034	Vietnam	walking	2020-04-11	48.27	-59.0
16035	Vietnam	driving	2020-04-12	45.59	-64.0
16036	Vietnam	walking	2020-04-12	46.31	-64.0
16037	Vietnam	driving	2020-04-13	46.78	-60.0
16038	Vietnam	walking	2020-04-13	47.65	-60.0
16039	Vietnam	driving	2020-04-14	48.57	-58.0
16040	Vietnam	walking	2020-04-14	48.82	-58.0
16041	Vietnam	driving	2020-04-15	54.63	-57.0
16042	Vietnam	walking	2020-04-15	53.42	-57.0
16043	Vietnam	driving	2020-04-16	56.93	-54.0
16044	Vietnam	walking	2020-04-16	56.38	-54.0
16045	Vietnam	driving	2020-04-17	57.54	-51.0
16046	Vietnam	walking	2020-04-17	57.43	-51.0

	grocery_and_pharmacy	parks	transit_stations	workplaces	residential
276	-3.0	-3.0	4.0	-1.0	0.0
277	-3.0	-3.0	4.0	-1.0	0.0
278	-6.0	-14.0	-2.0	-4.0	2.0
279	-6.0	-14.0	-2.0	-4.0	2.0
280	-8.0	-23.0	-2.0	7.0	1.0
281	-8.0	-23.0	-2.0	7.0	1.0
282	3.0	-2.0	9.0	9.0	-1.0
283	3.0	-2.0	9.0	9.0	-1.0
284	0.0	10.0	8.0	11.0	-2.0
285	0.0	10.0	8.0	11.0	-2.0
286	0.0	-8.0	7.0	9.0	-1.0
287	0.0	-8.0	7.0	9.0	-1.0
288	4.0	-9.0	9.0	9.0	-1.0

289	4.0	-9.0	9.0	9.0	-1.0
290	5.0	4.0	7.0	-1.0	0.0
291	5.0	4.0	7.0	-1.0	0.0
292	8.0	18.0	5.0	-2.0	0.0
293	8.0	18.0	5.0	-2.0	0.0
294	-16.0	26.0	-27.0	-54.0	6.0
295	-16.0	26.0	-27.0	-54.0	6.0
296	-17.0	1.0	-28.0	-50.0	7.0
297	-17.0	1.0	-28.0	-50.0	7.0
298	2.0	5.0	9.0	13.0	-2.0
299	2.0	5.0	9.0	13.0	-2.0
300	0.0	-5.0	8.0	12.0	0.0
301	0.0	-5.0	8.0	12.0	0.0
302	2.0	-8.0	8.0	13.0	-1.0
303	2.0	-8.0	8.0	13.0	-1.0
304	5.0	-8.0	6.0	2.0	1.0
305	5.0	-8.0	6.0	2.0	1.0
...	...	...	...	...	...
16017	-43.0	-43.0	-64.0	-36.0	22.0
16018	-43.0	-43.0	-64.0	-36.0	22.0
16019	-42.0	-49.0	-64.0	-33.0	23.0
16020	-42.0	-49.0	-64.0	-33.0	23.0
16021	-44.0	-47.0	-67.0	-32.0	22.0
16022	-44.0	-47.0	-67.0	-32.0	22.0
16023	-36.0	-38.0	-61.0	-25.0	20.0
16024	-36.0	-38.0	-61.0	-25.0	20.0
16025	-34.0	-38.0	-59.0	-23.0	19.0
16026	-34.0	-38.0	-59.0	-23.0	19.0
16027	-39.0	-39.0	-60.0	-23.0	17.0
16028	-39.0	-39.0	-60.0	-23.0	17.0
16029	-38.0	-39.0	-60.0	-24.0	15.0
16030	-38.0	-39.0	-60.0	-24.0	15.0
16031	-36.0	-38.0	-59.0	-30.0	20.0
16032	-36.0	-38.0	-59.0	-30.0	20.0
16033	-33.0	-43.0	-58.0	-29.0	21.0
16034	-33.0	-43.0	-58.0	-29.0	21.0
16035	-40.0	-46.0	-64.0	-30.0	20.0
16036	-40.0	-46.0	-64.0	-30.0	20.0
16037	-37.0	-40.0	-61.0	-23.0	20.0
16038	-37.0	-40.0	-61.0	-23.0	20.0
16039	-31.0	-36.0	-56.0	-19.0	18.0
16040	-31.0	-36.0	-56.0	-19.0	18.0
16041	-34.0	-37.0	-57.0	-19.0	16.0
16042	-34.0	-37.0	-57.0	-19.0	16.0
16043	-30.0	-34.0	-54.0	-16.0	13.0
16044	-30.0	-34.0	-54.0	-16.0	13.0
16045	-30.0	-35.0	-54.0	-25.0	18.0

```
16046                -30.0  -35.0             -54.0       -25.0          18.0

[8169 rows x 10 columns]
```

[8]:
```
# Now, let's add COVID data to the mobility Dataframe

covid_df = covid_df.drop(columns=[" Country_code", " WHO_region"]) # delete␣
 ↪extraneous columns
covid_df = covid_df.rename(columns={"Date_reported": "date", " Country":␣
 ↪"country"}) # Reformat column names to match mobility df

mobility_df = pd.merge(mobility_df, covid_df, how="outer", on=["country",␣
 ↪"date"])
mobility_df = mobility_df.dropna() # Remove instances without both COVID and␣
 ↪mobility data
mobility_df
```

[8]:

	country	transportation_type	date	value	retail	\
310	Argentina	driving	2020-03-03	88.90	0.0	
311	Argentina	walking	2020-03-03	104.39	0.0	
312	Argentina	driving	2020-03-04	92.74	-1.0	
313	Argentina	walking	2020-03-04	108.23	-1.0	
314	Argentina	driving	2020-03-05	97.45	1.0	
315	Argentina	walking	2020-03-05	113.92	1.0	
316	Argentina	driving	2020-03-06	118.32	6.0	
317	Argentina	walking	2020-03-06	126.70	6.0	
318	Argentina	driving	2020-03-07	122.78	5.0	
319	Argentina	walking	2020-03-07	112.81	5.0	
320	Argentina	driving	2020-03-08	80.36	3.0	
321	Argentina	walking	2020-03-08	70.94	3.0	
322	Argentina	driving	2020-03-09	89.28	1.0	
323	Argentina	walking	2020-03-09	109.38	1.0	
324	Argentina	driving	2020-03-10	89.29	3.0	
325	Argentina	walking	2020-03-10	109.62	3.0	
326	Argentina	driving	2020-03-11	82.73	-16.0	
327	Argentina	walking	2020-03-11	83.06	-16.0	
328	Argentina	driving	2020-03-12	86.67	-4.0	
329	Argentina	walking	2020-03-12	106.85	-4.0	
330	Argentina	driving	2020-03-13	98.75	-5.0	
331	Argentina	walking	2020-03-13	104.31	-5.0	
332	Argentina	driving	2020-03-14	84.77	-19.0	
333	Argentina	walking	2020-03-14	70.31	-19.0	
334	Argentina	driving	2020-03-15	47.70	-29.0	
335	Argentina	walking	2020-03-15	43.64	-29.0	
336	Argentina	driving	2020-03-16	53.57	-27.0	
337	Argentina	walking	2020-03-16	51.55	-27.0	
338	Argentina	driving	2020-03-17	45.44	-41.0	

339	Argentina	walking	2020-03-17	35.57	-41.0
...	...	...	...	...	...
15807	Uruguay	driving	2020-04-03	38.33	-53.0
15808	Uruguay	walking	2020-04-03	27.36	-53.0
15809	Uruguay	driving	2020-04-04	31.06	-58.0
15810	Uruguay	walking	2020-04-04	23.59	-58.0
15811	Uruguay	driving	2020-04-05	20.06	-71.0
15812	Uruguay	walking	2020-04-05	14.91	-71.0
15813	Uruguay	driving	2020-04-06	31.53	-55.0
15814	Uruguay	walking	2020-04-06	21.12	-55.0
15815	Uruguay	driving	2020-04-07	30.47	-56.0
15816	Uruguay	walking	2020-04-07	23.67	-56.0
15817	Uruguay	driving	2020-04-08	31.66	-55.0
15818	Uruguay	walking	2020-04-08	25.73	-55.0
15819	Uruguay	driving	2020-04-09	30.94	-63.0
15820	Uruguay	walking	2020-04-09	22.53	-63.0
15821	Uruguay	driving	2020-04-10	27.61	-76.0
15822	Uruguay	walking	2020-04-10	19.33	-76.0
15823	Uruguay	driving	2020-04-11	32.85	-68.0
15824	Uruguay	walking	2020-04-11	23.64	-68.0
15825	Uruguay	driving	2020-04-12	24.94	-71.0
15826	Uruguay	walking	2020-04-12	14.24	-71.0
15827	Uruguay	driving	2020-04-13	33.68	-47.0
15828	Uruguay	walking	2020-04-13	25.57	-47.0
15829	Uruguay	driving	2020-04-14	28.74	-60.0
15830	Uruguay	walking	2020-04-14	19.77	-60.0
15831	Uruguay	driving	2020-04-15	33.16	-52.0
15832	Uruguay	walking	2020-04-15	23.05	-52.0
15833	Uruguay	driving	2020-04-16	36.07	-53.0
15834	Uruguay	walking	2020-04-16	26.22	-53.0
15835	Uruguay	driving	2020-04-17	45.24	-52.0
15836	Uruguay	walking	2020-04-17	30.91	-52.0

	grocery_and_pharmacy	parks	transit_stations	workplaces	residential	\
310	5.0	-6.0	8.0	16.0	-1.0	
311	5.0	-6.0	8.0	16.0	-1.0	
312	3.0	-1.0	8.0	18.0	-2.0	
313	3.0	-1.0	8.0	18.0	-2.0	
314	4.0	-9.0	10.0	18.0	-1.0	
315	4.0	-9.0	10.0	18.0	-1.0	
316	8.0	-7.0	12.0	18.0	-2.0	
317	8.0	-7.0	12.0	18.0	-2.0	
318	7.0	-10.0	8.0	3.0	1.0	
319	7.0	-10.0	8.0	3.0	1.0	
320	4.0	-12.0	3.0	-2.0	2.0	
321	4.0	-12.0	3.0	-2.0	2.0	
322	5.0	-9.0	13.0	15.0	-1.0	

323	5.0	-9.0	13.0	15.0	-1.0
324	8.0	-9.0	14.0	18.0	-2.0
325	8.0	-9.0	14.0	18.0	-2.0
326	-7.0	-24.0	-6.0	15.0	0.0
327	-7.0	-24.0	-6.0	15.0	0.0
328	8.0	-16.0	10.0	18.0	-1.0
329	8.0	-16.0	10.0	18.0	-1.0
330	10.0	-17.0	7.0	18.0	-1.0
331	10.0	-17.0	7.0	18.0	-1.0
332	5.0	-40.0	-19.0	-4.0	7.0
333	5.0	-40.0	-19.0	-4.0	7.0
334	17.0	-44.0	-17.0	-10.0	9.0
335	17.0	-44.0	-17.0	-10.0	9.0
336	8.0	-39.0	-17.0	1.0	6.0
337	8.0	-39.0	-17.0	1.0	6.0
338	-11.0	-54.0	-32.0	-12.0	11.0
339	-11.0	-54.0	-32.0	-12.0	11.0
...	...	...	...	...	...
15807	-20.0	-66.0	-52.0	-39.0	21.0
15808	-20.0	-66.0	-52.0	-39.0	21.0
15809	-24.0	-71.0	-57.0	-32.0	19.0
15810	-24.0	-71.0	-57.0	-32.0	19.0
15811	-36.0	-79.0	-68.0	-35.0	21.0
15812	-36.0	-79.0	-68.0	-35.0	21.0
15813	-26.0	-69.0	-55.0	-48.0	21.0
15814	-26.0	-69.0	-55.0	-48.0	21.0
15815	-25.0	-70.0	-55.0	-50.0	21.0
15816	-25.0	-70.0	-55.0	-50.0	21.0
15817	-22.0	-70.0	-54.0	-50.0	20.0
15818	-22.0	-70.0	-54.0	-50.0	20.0
15819	-26.0	-71.0	-60.0	-58.0	23.0
15820	-26.0	-71.0	-60.0	-58.0	23.0
15821	-42.0	-77.0	-70.0	-67.0	28.0
15822	-42.0	-77.0	-70.0	-67.0	28.0
15823	-29.0	-75.0	-63.0	-43.0	20.0
15824	-29.0	-75.0	-63.0	-43.0	20.0
15825	-32.0	-79.0	-66.0	-32.0	20.0
15826	-32.0	-79.0	-66.0	-32.0	20.0
15827	-26.0	-66.0	-49.0	-34.0	18.0
15828	-26.0	-66.0	-49.0	-34.0	18.0
15829	-37.0	-74.0	-60.0	-39.0	20.0
15830	-37.0	-74.0	-60.0	-39.0	20.0
15831	-25.0	-69.0	-51.0	-36.0	18.0
15832	-25.0	-69.0	-51.0	-36.0	18.0
15833	-28.0	-65.0	-49.0	-34.0	18.0
15834	-28.0	-65.0	-49.0	-34.0	18.0
15835	-26.0	-66.0	-49.0	-32.0	19.0

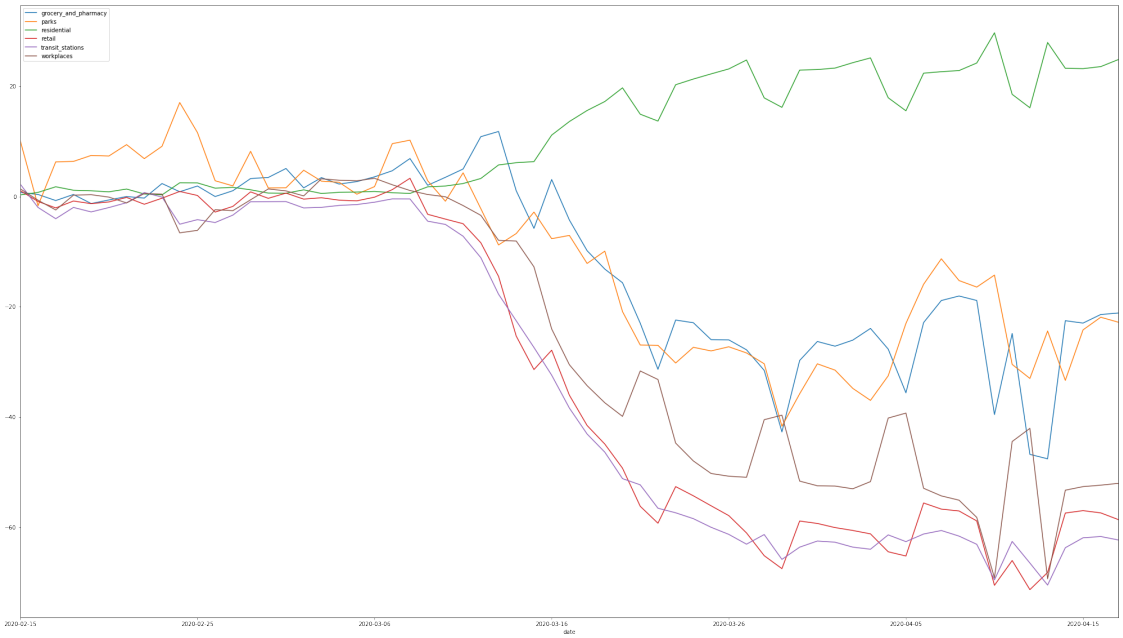
15836 -26.0 -66.0 -49.0 -32.0 19.0

	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
310	1.0	1.0	0.0	0.0
311	1.0	1.0	0.0	0.0
312	0.0	1.0	0.0	0.0
313	0.0	1.0	0.0	0.0
314	0.0	1.0	0.0	0.0
315	0.0	1.0	0.0	0.0
316	1.0	2.0	0.0	0.0
317	1.0	2.0	0.0	0.0
318	7.0	9.0	1.0	1.0
319	7.0	9.0	1.0	1.0
320	3.0	12.0	0.0	1.0
321	3.0	12.0	0.0	1.0
322	0.0	12.0	0.0	1.0
323	0.0	12.0	0.0	1.0
324	0.0	12.0	0.0	1.0
325	0.0	12.0	0.0	1.0
326	5.0	17.0	0.0	1.0
327	5.0	17.0	0.0	1.0
328	2.0	19.0	0.0	1.0
329	2.0	19.0	0.0	1.0
330	12.0	31.0	0.0	1.0
331	12.0	31.0	0.0	1.0
332	14.0	45.0	1.0	2.0
333	14.0	45.0	1.0	2.0
334	0.0	45.0	0.0	2.0
335	0.0	45.0	0.0	2.0
336	11.0	56.0	0.0	2.0
337	11.0	56.0	0.0	2.0
338	9.0	65.0	0.0	2.0
339	9.0	65.0	0.0	2.0
...	...	...	...	...
15807	12.0	350.0	2.0	4.0
15808	12.0	350.0	2.0	4.0
15809	19.0	369.0	0.0	4.0
15810	19.0	369.0	0.0	4.0
15811	17.0	386.0	0.0	4.0
15812	17.0	386.0	0.0	4.0
15813	14.0	400.0	1.0	5.0
15814	14.0	400.0	1.0	5.0
15815	6.0	406.0	1.0	6.0
15816	6.0	406.0	1.0	6.0
15817	9.0	415.0	0.0	6.0
15818	9.0	415.0	0.0	6.0
15819	9.0	424.0	1.0	7.0

```
15820        9.0             424.0          1.0                7.0
15821       22.0             446.0          0.0                7.0
15822       22.0             446.0          0.0                7.0
15823        7.0             453.0          0.0                7.0
15824        7.0             453.0          0.0                7.0
15825       12.0             465.0          0.0                7.0
15826       12.0             465.0          0.0                7.0
15827        7.0             472.0          0.0                7.0
15828        7.0             472.0          0.0                7.0
15829       11.0             483.0          0.0                7.0
15830       11.0             483.0          0.0                7.0
15831        0.0             483.0          1.0                8.0
15832        0.0             483.0          1.0                8.0
15833        9.0             492.0          0.0                8.0
15834        9.0             492.0          0.0                8.0
15835        1.0             493.0          1.0                9.0
15836        1.0             493.0          1.0                9.0

[6470 rows x 14 columns]
```

[9]:
```
# Just for reference, let's get a simple visualization of the mobility data over␣
 ↪time

mobility_date = mobility_df.sort_index(level=["date"])
mobility_date['retail'] = mobility_date['retail'].astype('double')
mobility_date['grocery_and_pharmacy'] = mobility_date['grocery_and_pharmacy'].
 ↪astype('double')
mobility_date['parks'] = mobility_date['parks'].astype('double')
mobility_date['transit_stations'] = mobility_date['transit_stations'].
 ↪astype('double')
mobility_date['workplaces'] = mobility_date['workplaces'].astype('double')
mobility_date['residential'] = mobility_date['residential'].astype('double')
mobility_date = mobility_date.groupby(['date']).mean()

# Let's remove the "value" column and COVID case/death columns so that we can␣
 ↪just see the different modes of mobility
mobility_date = mobility_date[mobility_date.columns.difference(['value', "␣
 ↪Cumulative_cases", " Cumulative_deaths", " New_cases", " New_deaths"])]



mobility_date.plot(figsize=(35,20))
```

[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a99cd0feb8>
```



[10]:
```
# Out of curiosity, let's get a list of countries included in the above␣
 ↪Dataframe, as well as the country count
countries_list = mobility_df["country"].unique()
print("Number of countries included: ", len(countries_list), "\n",␣
 ↪countries_list)
```

```
Number of countries included:  49
 ['Argentina' 'Australia' 'Austria' 'Belgium' 'Brazil' 'Bulgaria'
 'Cambodia' 'Canada' 'Chile' 'Colombia' 'Croatia' 'Denmark' 'Egypt'
 'Estonia' 'Finland' 'France' 'Germany' 'Greece' 'Hungary' 'India'
 'Indonesia' 'Ireland' 'Israel' 'Italy' 'Japan' 'Latvia' 'Lithuania'
 'Luxembourg' 'Malaysia' 'Mexico' 'Netherlands' 'New Zealand' 'Norway'
 'Philippines' 'Poland' 'Portugal' 'Romania' 'Saudi Arabia' 'Singapore'
 'Slovakia' 'Slovenia' 'South Africa' 'Spain' 'Sweden' 'Switzerland'
 'Thailand' 'Turkey' 'United Arab Emirates' 'Uruguay']
```

[11]:
```
# And let's see the date range we are working with now.
print("Start date: ", mobility_df.date.min(), "\nEnd date: ", mobility_df.date.
 ↪max())
```

```
Start date:  2020-02-15
End date:  2020-04-17
```

[12]:
```
# Let's see that restaurant dataframe again

restaurant_df
```

[12]:

region_type	country	percent_yoy_change	date
countries	Global	-1	2020-02-18
countries	Global	3	2020-02-19
countries	Global	-1	2020-02-20
countries	Global	-2	2020-02-21
countries	Global	1	2020-02-22
countries	Global	4	2020-02-23
countries	Global	1	2020-02-24
countries	Global	1	2020-02-25
countries	Global	-2	2020-02-26
countries	Global	-3	2020-02-27
countries	Global	-1	2020-02-28
countries	Global	1	2020-02-29
countries	Global	0	2020-03-01
countries	Global	-8	2020-03-02
countries	Global	-9	2020-03-03
countries	Global	-6	2020-03-04
countries	Global	-7	2020-03-05
countries	Global	-7	2020-03-06
countries	Global	-4	2020-03-07
countries	Global	-3	2020-03-08
countries	Global	-14	2020-03-09
countries	Global	-18	2020-03-10
countries	Global	-19	2020-03-11
countries	Global	-28	2020-03-12
countries	Global	-36	2020-03-13
countries	Global	-40	2020-03-14
countries	Global	-47	2020-03-15
countries	Global	-56	2020-03-16
countries	Global	-83	2020-03-17
countries	Global	-89	2020-03-18
...	...	...	...
countries	United States	-67	2020-06-15
countries	United States	-66	2020-06-16
countries	United States	-65	2020-06-17
countries	United States	-64	2020-06-18
countries	United States	-59	2020-06-19
countries	United States	-58	2020-06-20
countries	United States	-41	2020-06-21
countries	United States	-66	2020-06-22
countries	United States	-65	2020-06-23
countries	United States	-63	2020-06-24
countries	United States	-61	2020-06-25
countries	United States	-57	2020-06-26
countries	United States	-57	2020-06-27
countries	United States	-60	2020-06-28

```
countries    United States                -65  2020-06-29
countries    United States                -62  2020-06-30
countries    United States                -64  2020-07-01
countries    United States                -49  2020-07-02
countries    United States                -57  2020-07-03
countries    United States                -72  2020-07-04
countries    United States                -59  2020-07-05
countries    United States                -63  2020-07-06
countries    United States                -64  2020-07-07
countries    United States                -65  2020-07-08
countries    United States                -62  2020-07-09
countries    United States                -62  2020-07-10
countries    United States                -57  2020-07-11
countries    United States                -59  2020-07-12
countries    United States                -66  2020-07-13
countries    United States                -65  2020-07-14

[1184 rows x 3 columns]
```

[13]:
```
# We only want data for specific countries, so let's get rid of all the "Global"␣
 ↪values, and also any rows with null values

restaurant_df.dropna() # drops any possible rows with null values
restaurant_df = restaurant_df[restaurant_df.country != 'Global'] # Only keeps␣
 ↪instances of countries, excludes global data instances

restaurant_df
```

[13]:

region_type	country	percent_yoy_change	date
countries	Australia	-3	2020-02-18
countries	Australia	-6	2020-02-19
countries	Australia	-3	2020-02-20
countries	Australia	-1	2020-02-21
countries	Australia	0	2020-02-22
countries	Australia	0	2020-02-23
countries	Australia	0	2020-02-24
countries	Australia	-2	2020-02-25
countries	Australia	-2	2020-02-26
countries	Australia	-7	2020-02-27
countries	Australia	0	2020-02-28
countries	Australia	-1	2020-02-29
countries	Australia	4	2020-03-01
countries	Australia	-12	2020-03-02
countries	Australia	-6	2020-03-03
countries	Australia	-12	2020-03-04
countries	Australia	-8	2020-03-05

```
countries        Australia                -6  2020-03-06
countries        Australia                -3  2020-03-07
countries        Australia                -4  2020-03-08
countries        Australia               -10  2020-03-09
countries        Australia                -4  2020-03-10
countries        Australia               -10  2020-03-11
countries        Australia               -12  2020-03-12
countries        Australia                -8  2020-03-13
countries        Australia               -11  2020-03-14
countries        Australia               -12  2020-03-15
countries        Australia               -30  2020-03-16
countries        Australia               -43  2020-03-17
countries        Australia               -50  2020-03-18
...                    ...               ...         ...
countries    United States               -67  2020-06-15
countries    United States               -66  2020-06-16
countries    United States               -65  2020-06-17
countries    United States               -64  2020-06-18
countries    United States               -59  2020-06-19
countries    United States               -58  2020-06-20
countries    United States               -41  2020-06-21
countries    United States               -66  2020-06-22
countries    United States               -65  2020-06-23
countries    United States               -63  2020-06-24
countries    United States               -61  2020-06-25
countries    United States               -57  2020-06-26
countries    United States               -57  2020-06-27
countries    United States               -60  2020-06-28
countries    United States               -65  2020-06-29
countries    United States               -62  2020-06-30
countries    United States               -64  2020-07-01
countries    United States               -49  2020-07-02
countries    United States               -57  2020-07-03
countries    United States               -72  2020-07-04
countries    United States               -59  2020-07-05
countries    United States               -63  2020-07-06
countries    United States               -64  2020-07-07
countries    United States               -65  2020-07-08
countries    United States               -62  2020-07-09
countries    United States               -62  2020-07-10
countries    United States               -57  2020-07-11
countries    United States               -59  2020-07-12
countries    United States               -66  2020-07-13
countries    United States               -65  2020-07-14

[1036 rows x 3 columns]
```

[14]:
```
# Now, just like with the mobility data, let's add in data on COVID cases

restaurant_df = pd.merge(restaurant_df, covid_df, how="outer", on=["country",␣
 ↪"date"])
restaurant_df = restaurant_df.dropna() # Remove instances without both COVID and␣
 ↪restaurant data
restaurant_df
```

[14]:

	country	percent_yoy_change	date	New_cases	Cumulative_cases
0	Australia	-3.0	2020-02-18	0.0	15.0
1	Australia	-6.0	2020-02-19	0.0	15.0
2	Australia	-3.0	2020-02-20	0.0	15.0
3	Australia	-1.0	2020-02-21	2.0	17.0
4	Australia	0.0	2020-02-22	4.0	21.0
5	Australia	0.0	2020-02-23	1.0	22.0
6	Australia	0.0	2020-02-24	0.0	22.0
7	Australia	-2.0	2020-02-25	1.0	23.0
8	Australia	-2.0	2020-02-26	0.0	23.0
9	Australia	-7.0	2020-02-27	0.0	23.0
10	Australia	0.0	2020-02-28	0.0	23.0
11	Australia	-1.0	2020-02-29	2.0	25.0
12	Australia	4.0	2020-03-01	2.0	27.0
13	Australia	-12.0	2020-03-02	6.0	33.0
14	Australia	-6.0	2020-03-03	0.0	33.0
15	Australia	-12.0	2020-03-04	0.0	33.0
16	Australia	-8.0	2020-03-05	24.0	57.0
17	Australia	-6.0	2020-03-06	5.0	62.0
18	Australia	-3.0	2020-03-07	8.0	70.0
19	Australia	-4.0	2020-03-08	7.0	77.0
20	Australia	-10.0	2020-03-09	15.0	92.0
21	Australia	-4.0	2020-03-10	0.0	92.0
22	Australia	-10.0	2020-03-11	30.0	122.0
23	Australia	-12.0	2020-03-12	18.0	140.0
24	Australia	-8.0	2020-03-13	49.0	189.0
25	Australia	-11.0	2020-03-14	8.0	197.0
26	Australia	-12.0	2020-03-15	101.0	298.0
27	Australia	-30.0	2020-03-16	38.0	336.0
28	Australia	-43.0	2020-03-17	78.0	414.0
29	Australia	-50.0	2020-03-18	96.0	510.0
..	...	...	...	...	...
708	Mexico	-91.0	2020-06-13	4790.0	133974.0
709	Mexico	-95.0	2020-06-14	5222.0	139196.0
710	Mexico	-91.0	2020-06-15	3494.0	142690.0
711	Mexico	-91.0	2020-06-16	4147.0	146837.0
712	Mexico	-91.0	2020-06-17	3427.0	150264.0
713	Mexico	-90.0	2020-06-18	4599.0	154863.0
714	Mexico	-87.0	2020-06-19	4930.0	159793.0

715	Mexico	-86.0	2020-06-20	5662.0	165455.0
716	Mexico	-83.0	2020-06-21	5030.0	170485.0
717	Mexico	-88.0	2020-06-22	4717.0	175202.0
718	Mexico	-88.0	2020-06-23	5343.0	180545.0
719	Mexico	-88.0	2020-06-24	4577.0	185122.0
720	Mexico	-87.0	2020-06-25	6288.0	191410.0
721	Mexico	-84.0	2020-06-26	5437.0	196847.0
722	Mexico	-84.0	2020-06-27	6104.0	202951.0
723	Mexico	-86.0	2020-06-28	5441.0	208392.0
724	Mexico	-87.0	2020-06-29	4410.0	212802.0
725	Mexico	-87.0	2020-06-30	4050.0	216852.0
726	Mexico	-82.0	2020-07-01	3805.0	220657.0
727	Mexico	-82.0	2020-07-02	5432.0	226089.0
728	Mexico	-79.0	2020-07-03	5681.0	231770.0
729	Mexico	-79.0	2020-07-04	6741.0	238511.0
730	Mexico	-78.0	2020-07-05	6740.0	245251.0
731	Mexico	-76.0	2020-07-06	6914.0	252165.0
732	Mexico	-79.0	2020-07-07	4683.0	256848.0
733	Mexico	-79.0	2020-07-08	4902.0	261750.0
734	Mexico	-78.0	2020-07-09	6258.0	268008.0
735	Mexico	-74.0	2020-07-10	6995.0	275003.0
736	Mexico	-75.0	2020-07-11	7280.0	282283.0
737	Mexico	-76.0	2020-07-12	6891.0	289174.0

	New_deaths	Cumulative_deaths
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0
5	0.0	0.0
6	0.0	0.0
7	0.0	0.0
8	0.0	0.0
9	0.0	0.0
10	0.0	0.0
11	0.0	0.0
12	1.0	1.0
13	0.0	1.0
14	0.0	1.0
15	0.0	1.0
16	1.0	2.0
17	0.0	2.0
18	0.0	2.0
19	1.0	3.0
20	0.0	3.0
21	0.0	3.0

```
22          0.0               3.0
23          0.0               3.0
24          0.0               3.0
25          0.0               3.0
26          2.0               5.0
27          0.0               5.0
28          0.0               5.0
29          1.0               6.0
..          ...               ...
708       587.0           15944.0
709       504.0           16448.0
710       424.0           16872.0
711       269.0           17141.0
712       439.0           17580.0
713       730.0           18310.0
714       770.0           19080.0
715       667.0           19747.0
716       647.0           20394.0
717       387.0           20781.0
718      1044.0           21825.0
719       759.0           22584.0
720       793.0           23377.0
721       947.0           24324.0
722       736.0           25060.0
723       719.0           25779.0
724       602.0           26381.0
725       267.0           26648.0
726       473.0           27121.0
727       648.0           27769.0
728       741.0           28510.0
729       679.0           29189.0
730       654.0           29843.0
731       523.0           30366.0
732       273.0           30639.0
733       480.0           31119.0
734       895.0           32014.0
735       782.0           32796.0
736       730.0           33526.0
737       665.0           34191.0

[708 rows x 7 columns]
```

[15]:
```
# Again, out of curiosity, let's find out which countries are included in this␣
↪data

restaurant_countries_list = restaurant_df["country"].unique()
```

```
print("Number of countries included: ", len(restaurant_countries_list), "\n",␣
 ↪restaurant_countries_list)
```

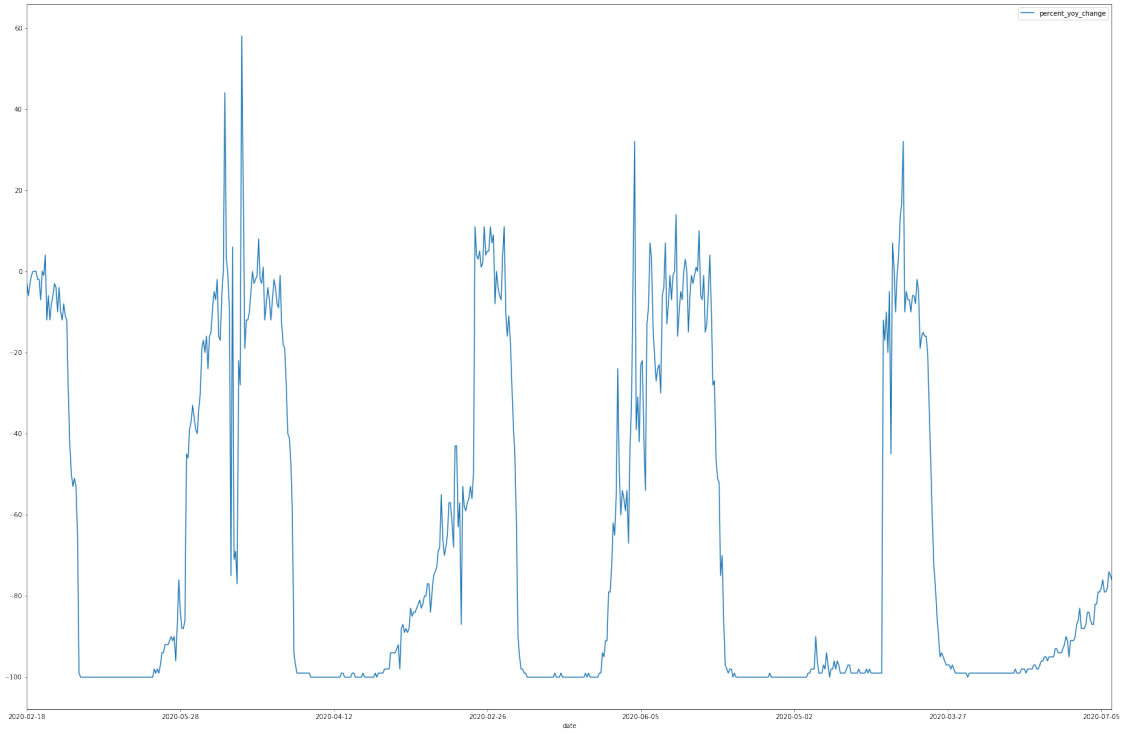
```
Number of countries included:  5
 ['Australia' 'Canada' 'Germany' 'Ireland' 'Mexico']
```

[16]:
```
# And the date range
print("Start date: ", restaurant_df.date.min(), "\nEnd date: ", restaurant_df.
 ↪date.max())
```

```
Start date:  2020-02-18
End date:  2020-07-12
```

[17]:
```
# Just for reference, let's look at the time series data for restaurant␣
 ↪performance across all countries

restaurant_df.plot("date", "percent_yoy_change", figsize=(30,20))
```

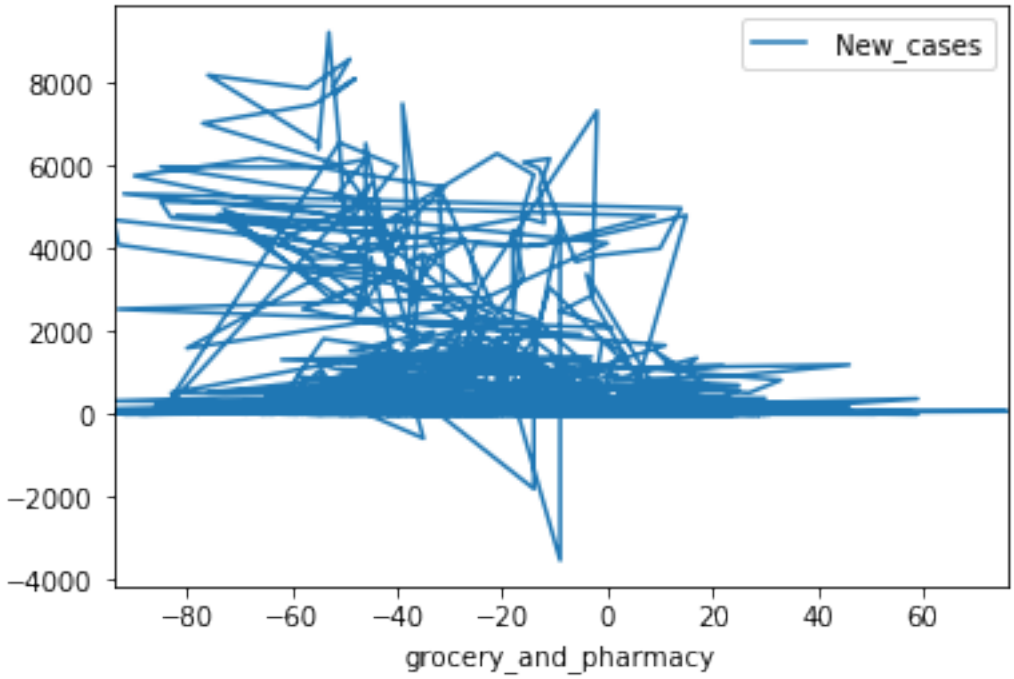
[17]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99cfcd860>



[18]:
```
# Let's start out plotting correlation between various types of mobility and new␣
 ↪cases of COVID-19
# Note that this is on a global scale (we can break it up by country later)
# First, plotting grocery and pharmacy trips
```

```
mobility_df.plot("grocery_and_pharmacy", " New_cases")
```

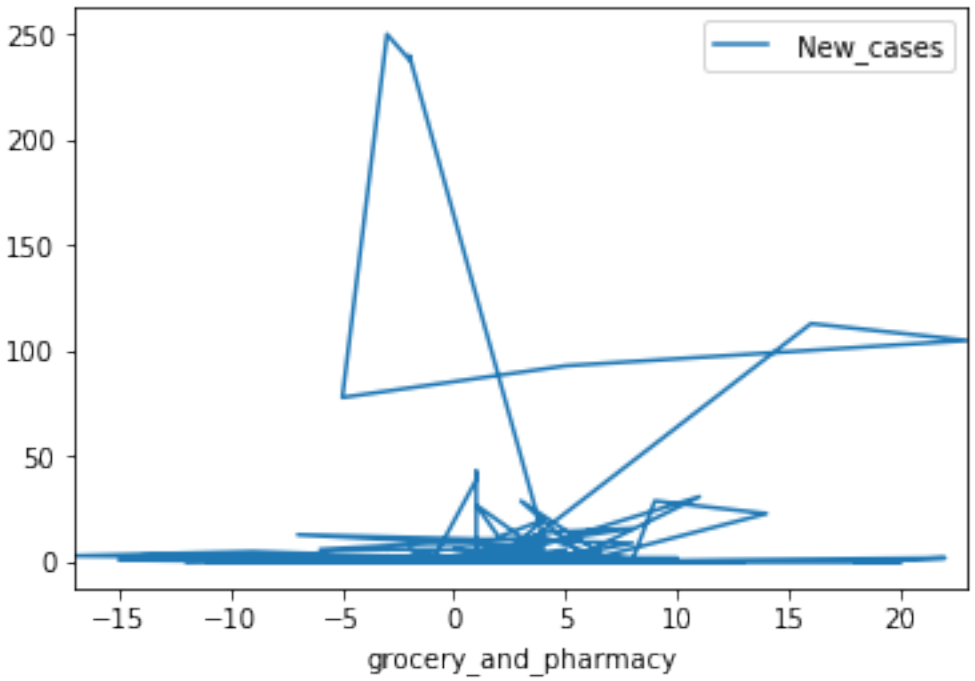
[18]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d0200b8>



[19]:
```
# Let's try breaking that down by month
feb_df = mobility_df[mobility_df["date"].str.contains("2020-02")]
march_df = mobility_df[mobility_df["date"].str.contains("2020-03")]
april_df = mobility_df[mobility_df["date"].str.contains("2020-04")]
```

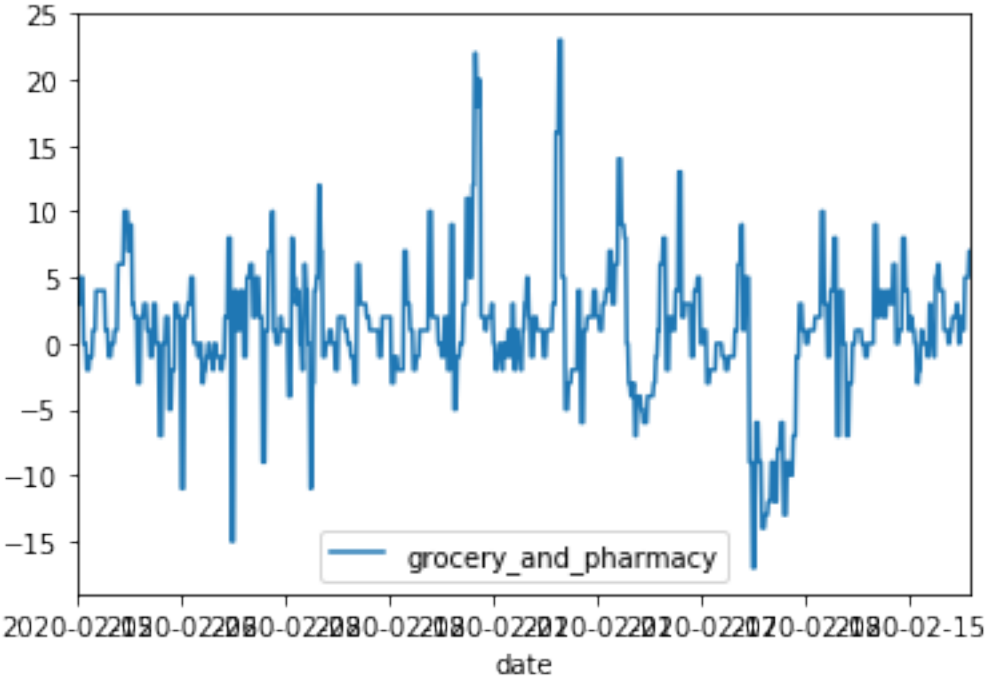
[20]:
```
feb_df.plot("grocery_and_pharmacy", " New_cases")
```

[20]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d08edd8>



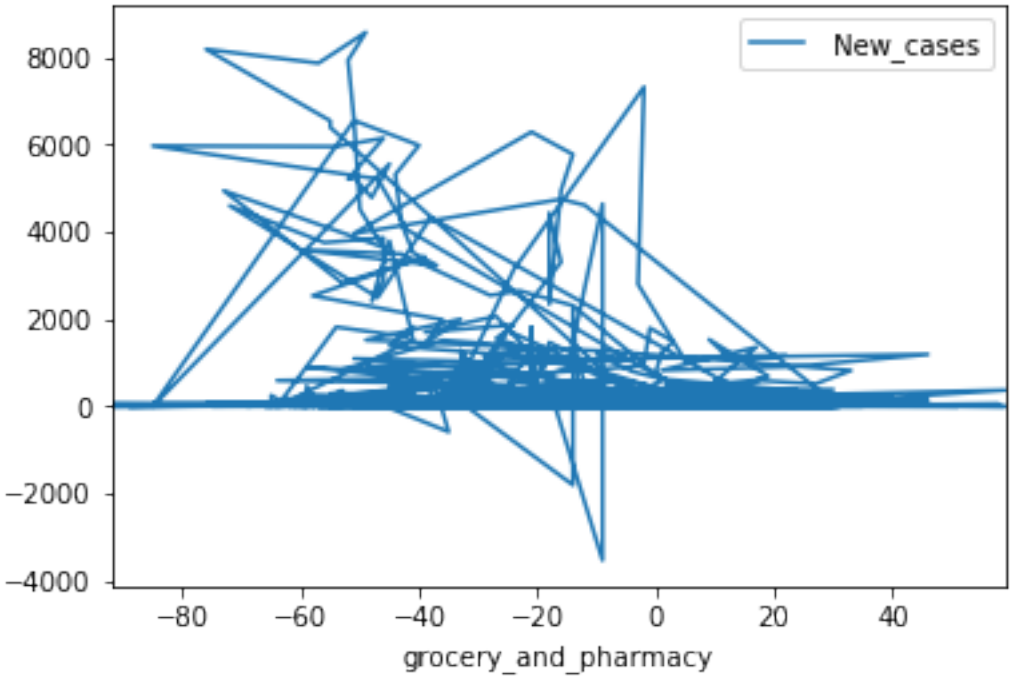
[21]:
```
# What did grocery/pharmacy trends look like overall in February?
feb_df.plot("date", "grocery_and_pharmacy")
```

[21]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d507908>



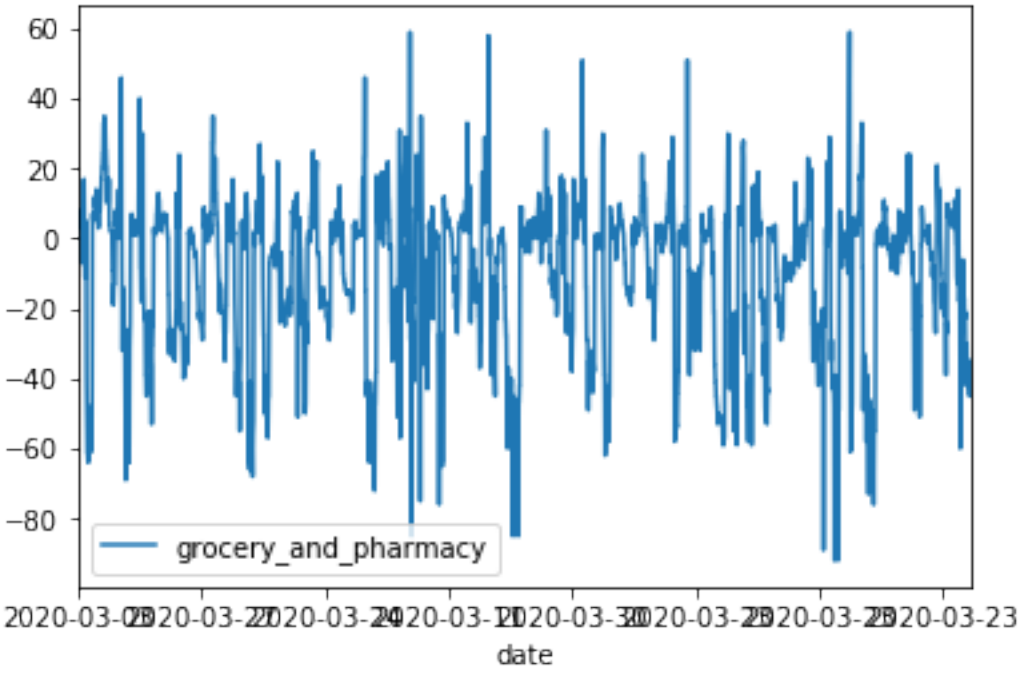
[22]:
```
march_df.plot("grocery_and_pharmacy", " New_cases")
```

[22]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d59ab70>



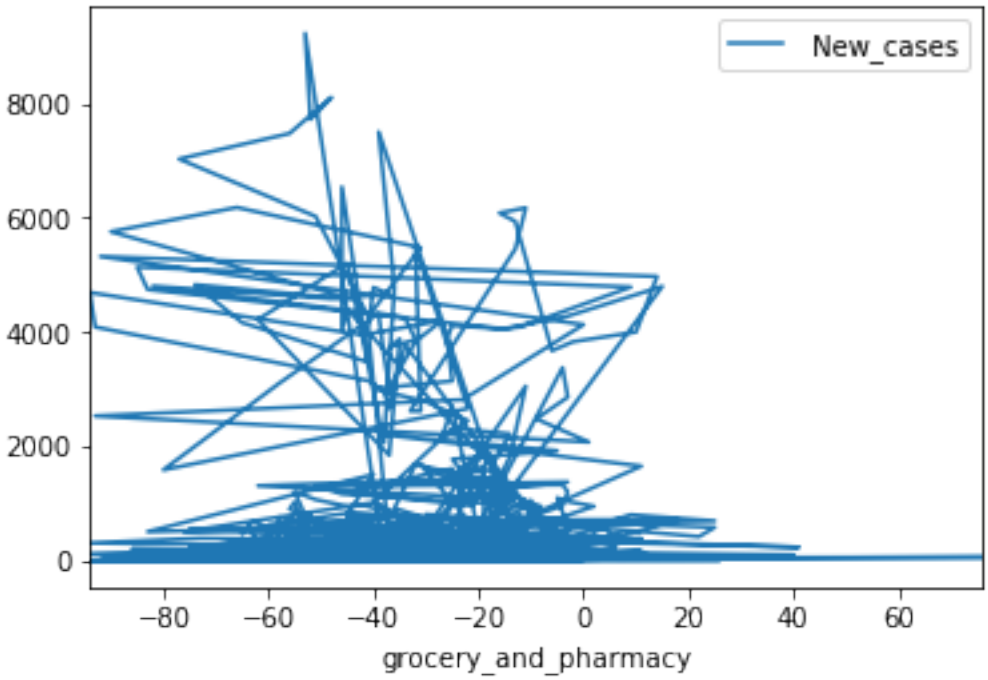
[23]:
```
march_df.plot("date", "grocery_and_pharmacy")
```

[23]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d59a2e8>



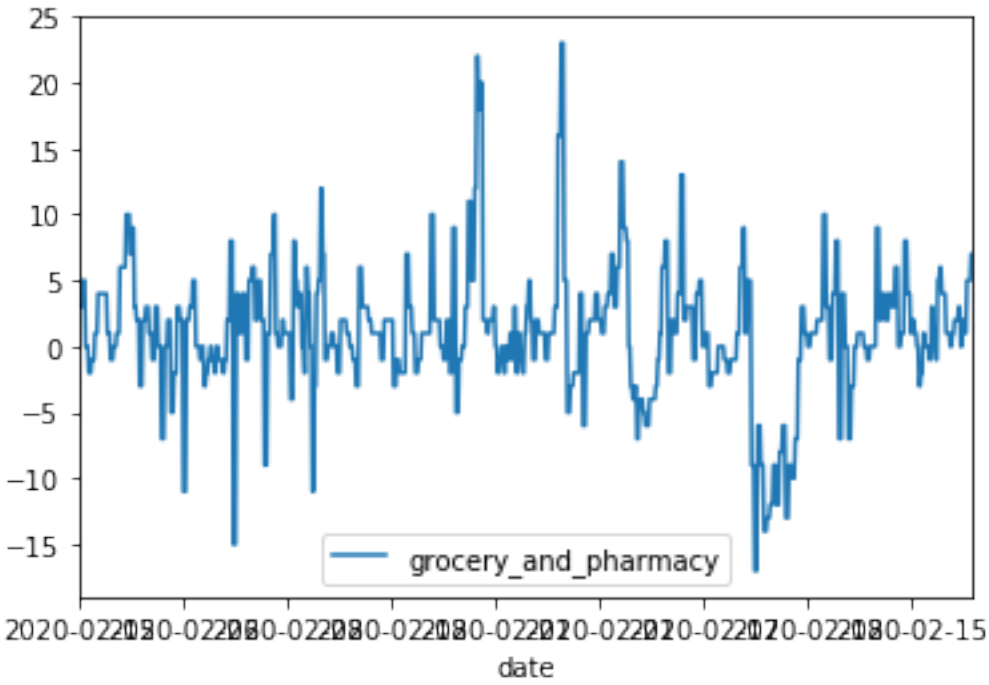
[24]:
```
april_df.plot("grocery_and_pharmacy", " New_cases")
```

[24]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d674c18>



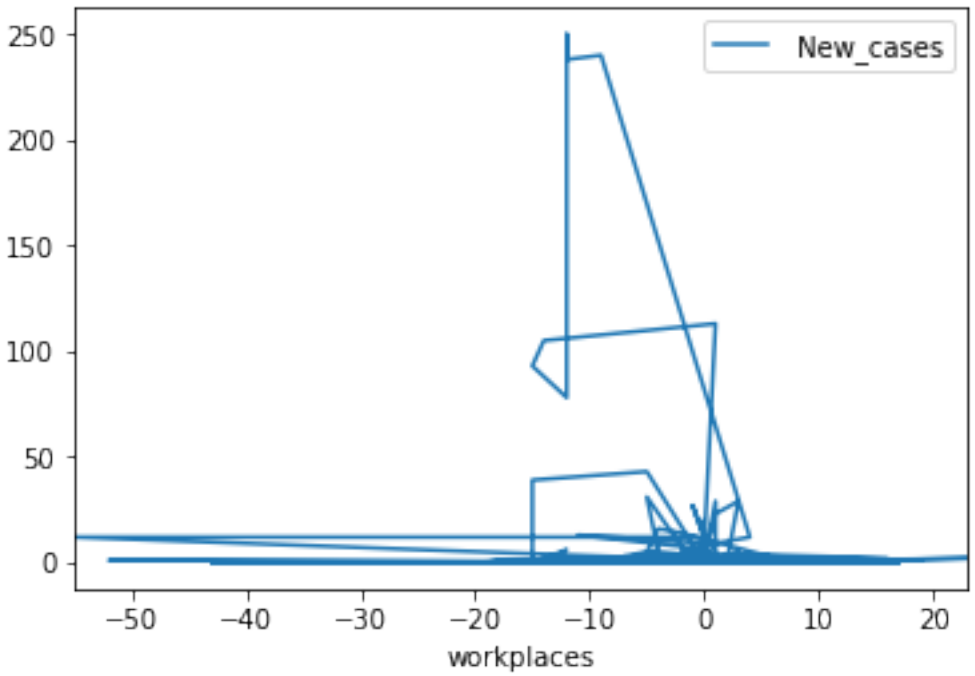
[25]:
```
feb_df.plot("date", "grocery_and_pharmacy")
```

[25]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d6e0748>



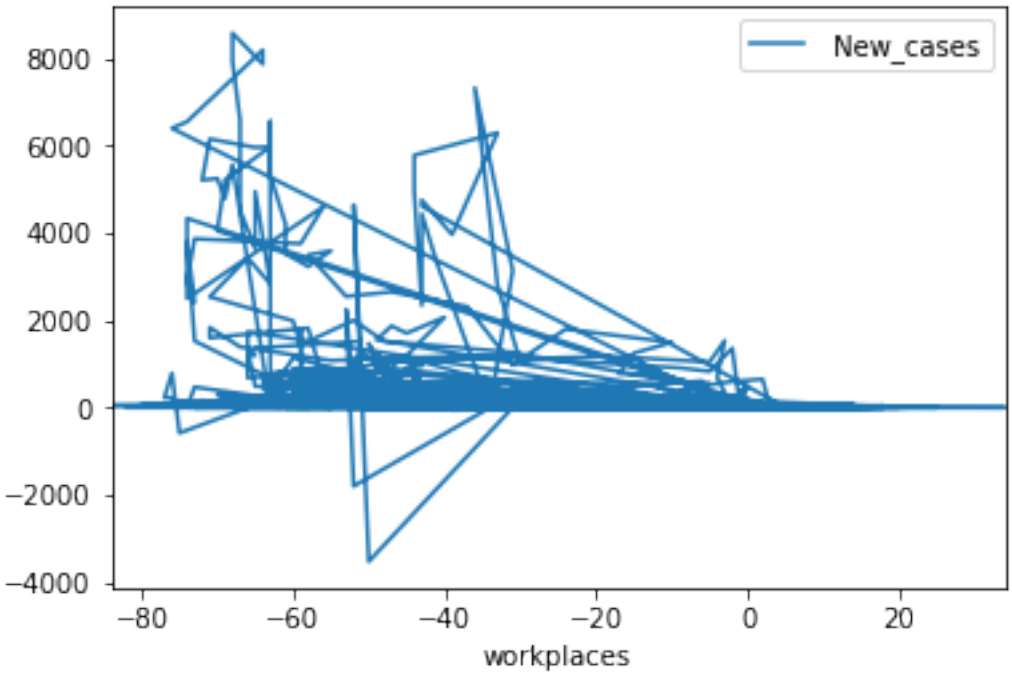
[26]:
```
# Let's try looking at workplace travel
feb_df.plot("workplaces", " New_cases")
```

[26]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d7551d0>



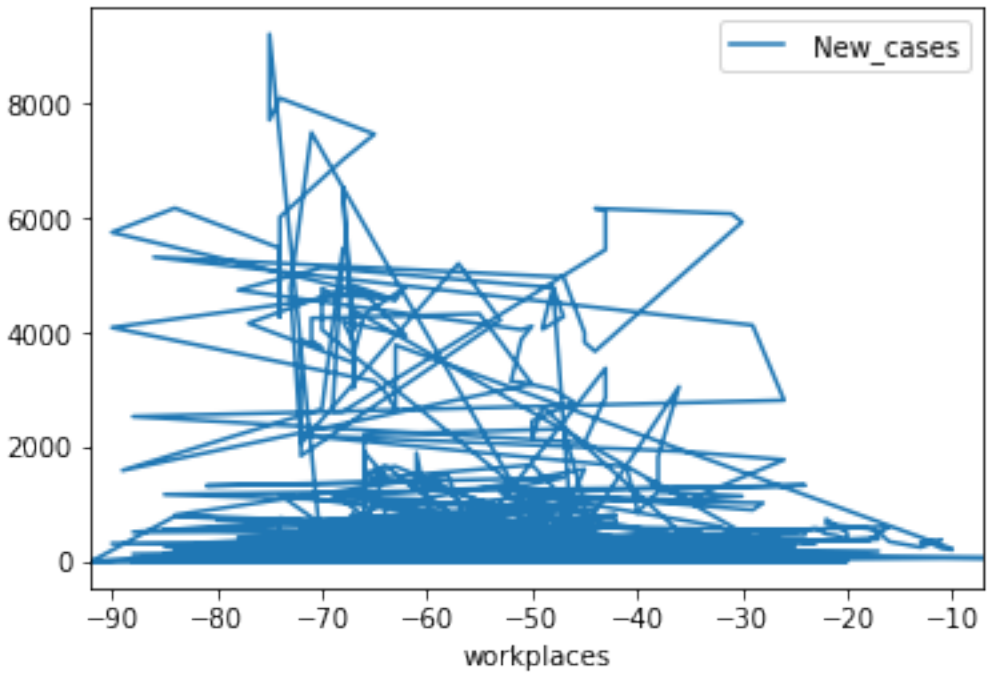
[27]:
```
march_df.plot("workplaces", " New_cases")
```

[27]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d7b79e8>



[28]:
```
april_df.plot("workplaces", " New_cases")
```

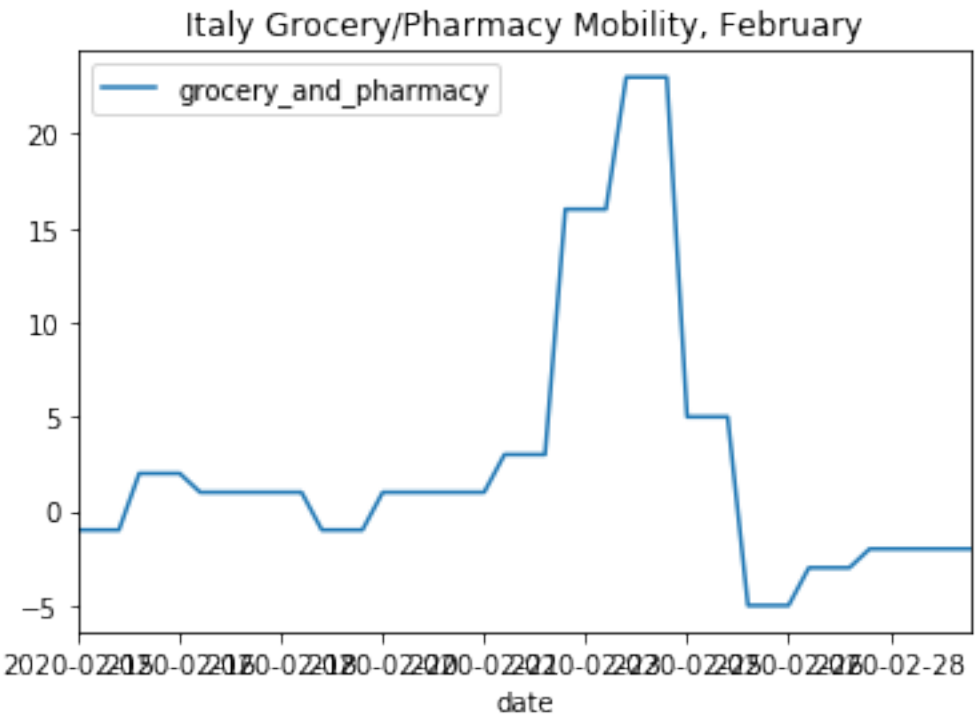
[28]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d813f60>



[29]:
```
# Maybe this will work better if I break it down by country as well. Let's use␣
 ↪Italy as an example.
italy_feb_df = feb_df[feb_df["country"].str.contains("Italy")]
italy_march_df = march_df[march_df["country"].str.contains("Italy")]
italy_april_df = april_df[april_df["country"].str.contains("Italy")]
```

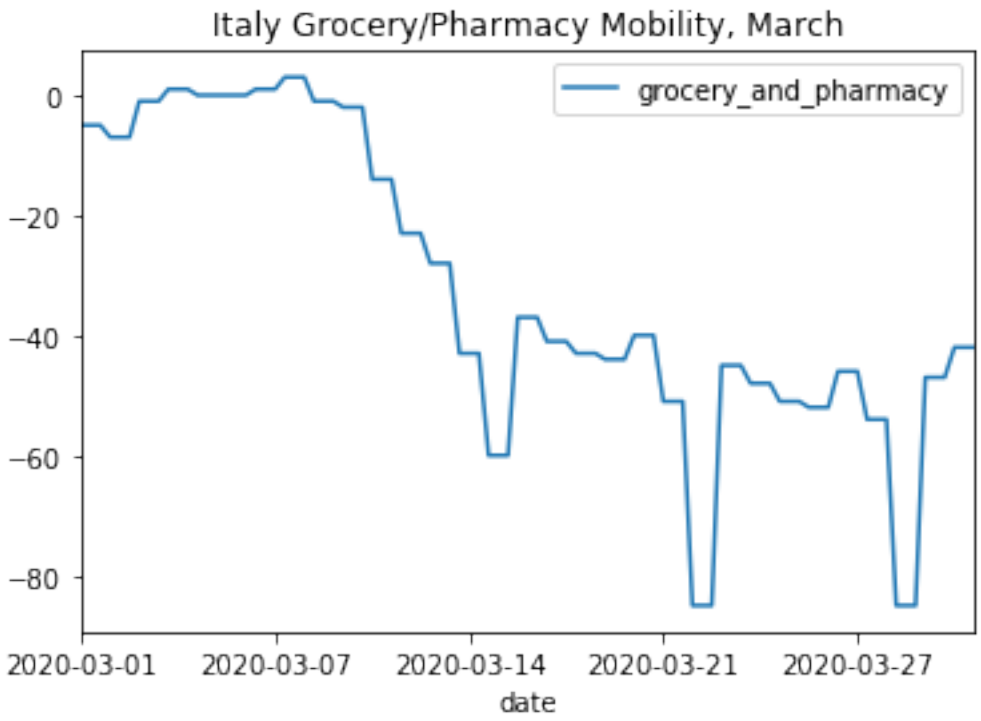
[30]:
```
# Let's see what the grocery/pharmacy trends look like over time now
italy_feb_df.plot("date", "grocery_and_pharmacy", title="Italy Grocery/Pharmacy␣
 ↪Mobility, February")
```

[30]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d86b5f8>



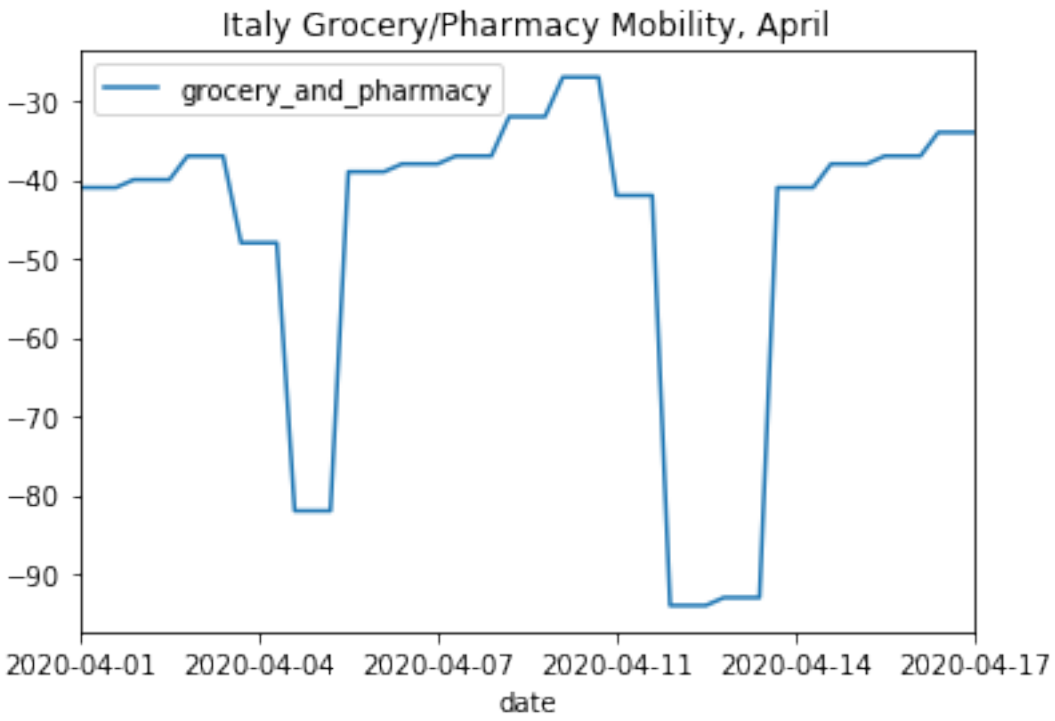
[31]:
```
italy_march_df.plot("date", "grocery_and_pharmacy", title="Italy Grocery/
↪Pharmacy Mobility, March")
```

[31]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d8ed320>



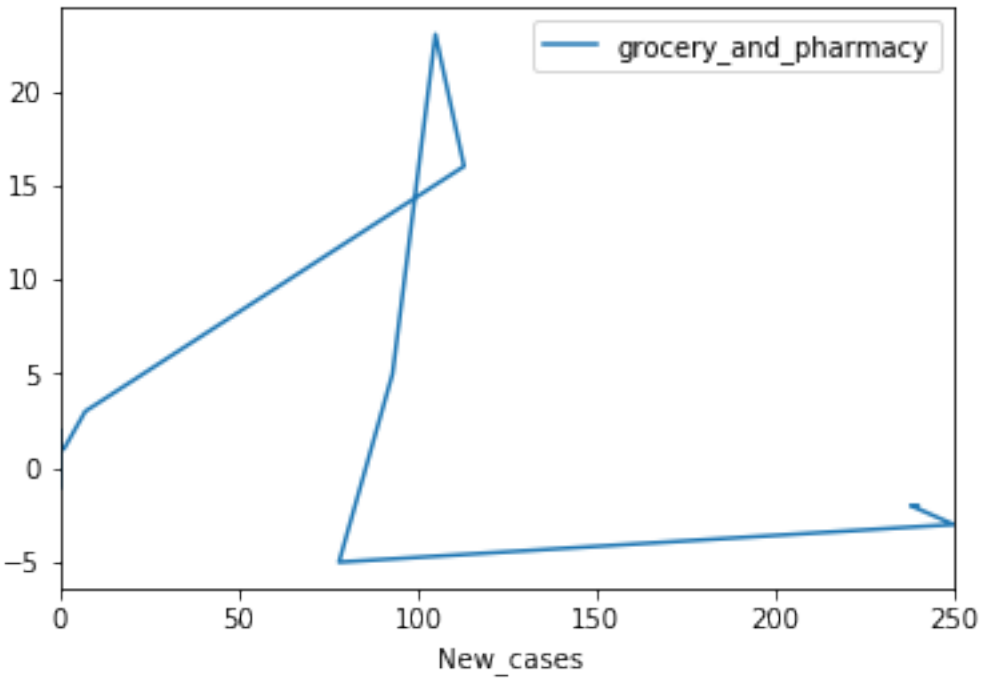
[32]:
```
italy_april_df.plot("date", "grocery_and_pharmacy", title="Italy Grocery/
↪Pharmacy Mobility, April")
```

[32]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d948630>



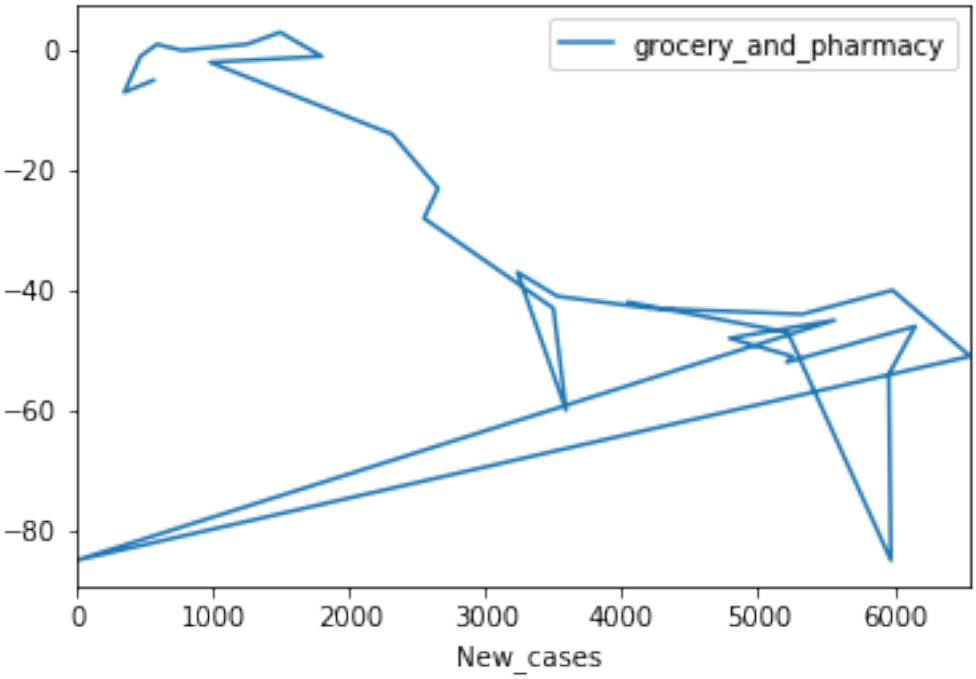
[33]:
```
# There are still multiple entries for each date, but this makes things easier␣
 ↪to see.
# Now let's look at correlation between grocery/pharmacy visits and new cases.
italy_feb_df.plot(" New_cases", "grocery_and_pharmacy")
```

[33]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d99b240>



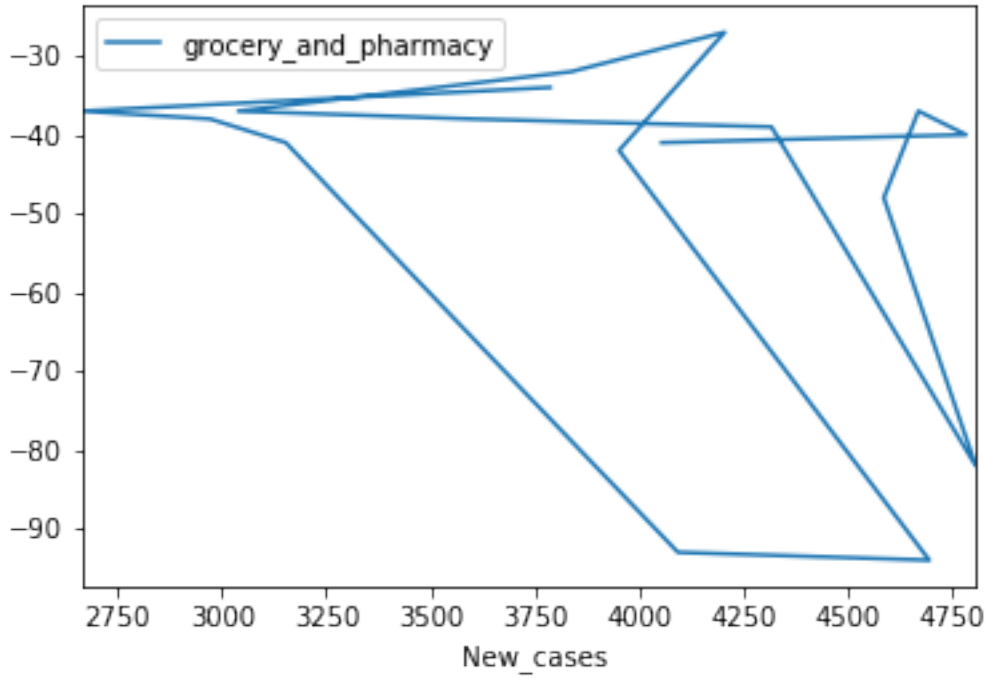
[34]:
```
italy_march_df.plot(" New_cases", "grocery_and_pharmacy")
```

[34]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99d9f98d0>



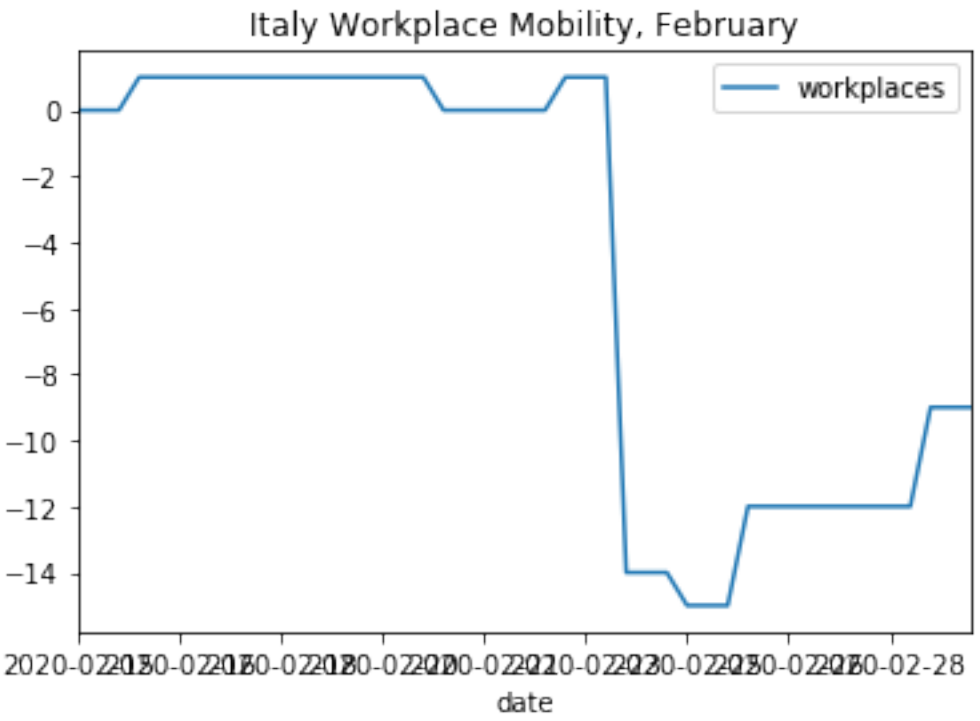
[35]:
```
italy_april_df.plot(" New_cases", "grocery_and_pharmacy")
```

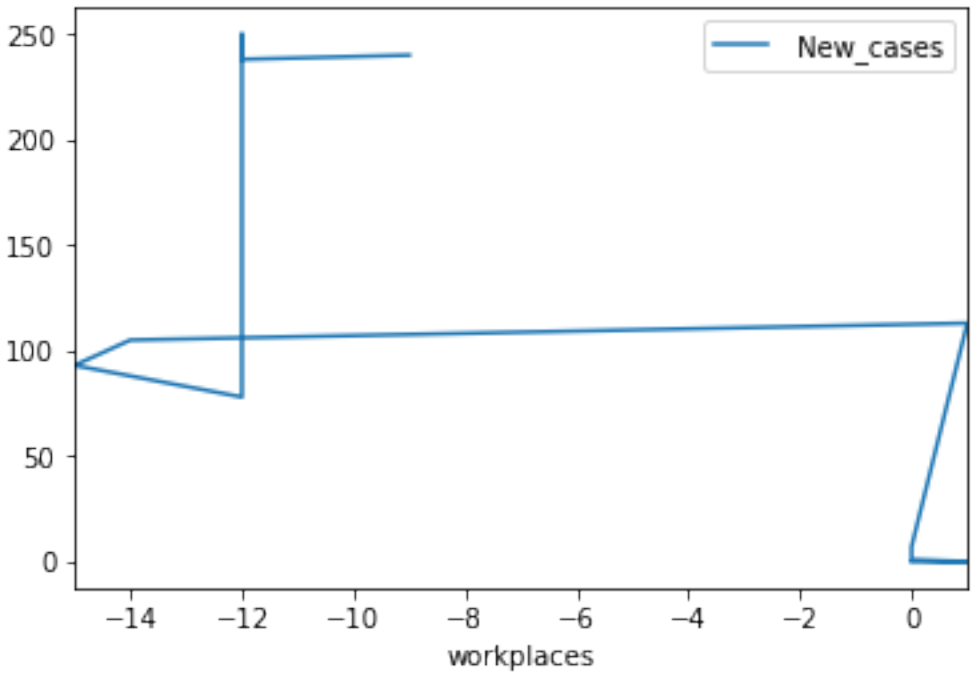
[35]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99da566a0>



[36]:
```
# Let's try workplaces
italy_feb_df.plot("date", "workplaces", title="Italy Workplace Mobility,␣
 ↪February")
italy_feb_df.plot("workplaces", " New_cases")
```

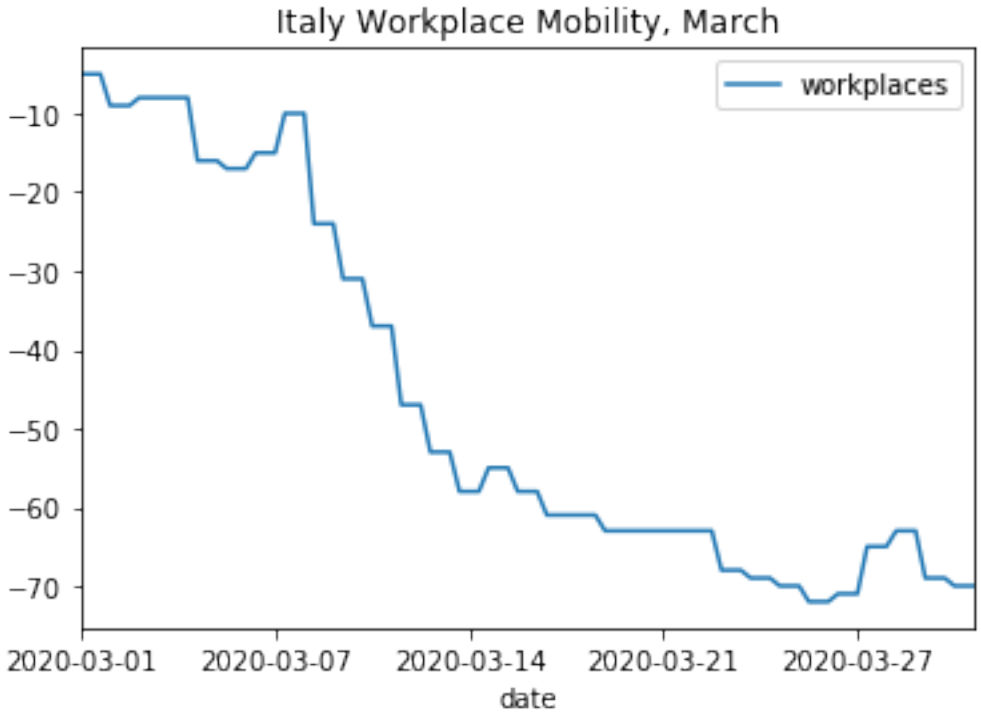
[36]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99db06128>

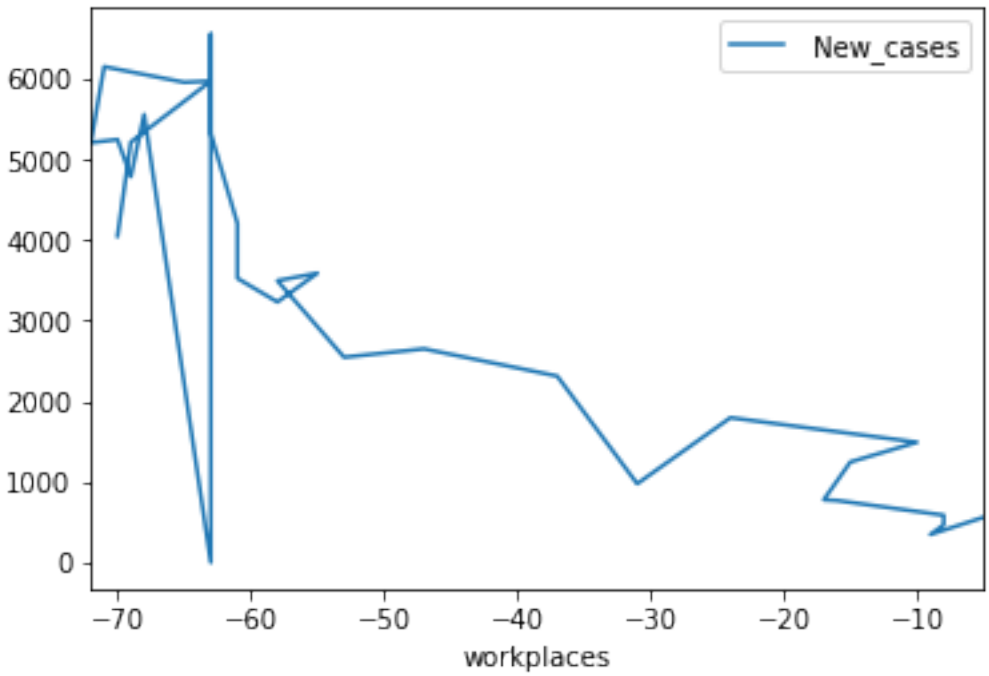

Italy Workplace Mobility, February
workplaces
0
−2
−4
−6
−8
−10
−12
−14
2020-02-28
date


New_cases
250
200
150
100
50
0
−14
−12
−10
−8
−6
−4
−2
0
workplaces

[37]:
```
italy_march_df.plot("date", "workplaces", title="Italy Workplace Mobility,␣
 ↪March")
italy_march_df.plot("workplaces", " New_cases")
```

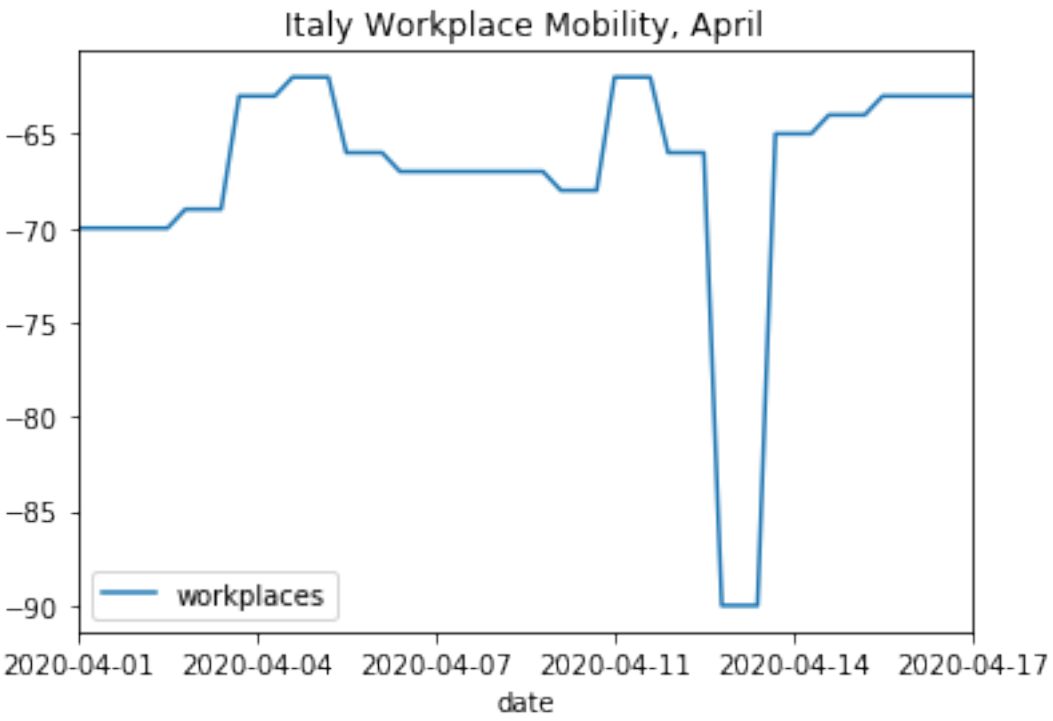
[37]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99db97da0>

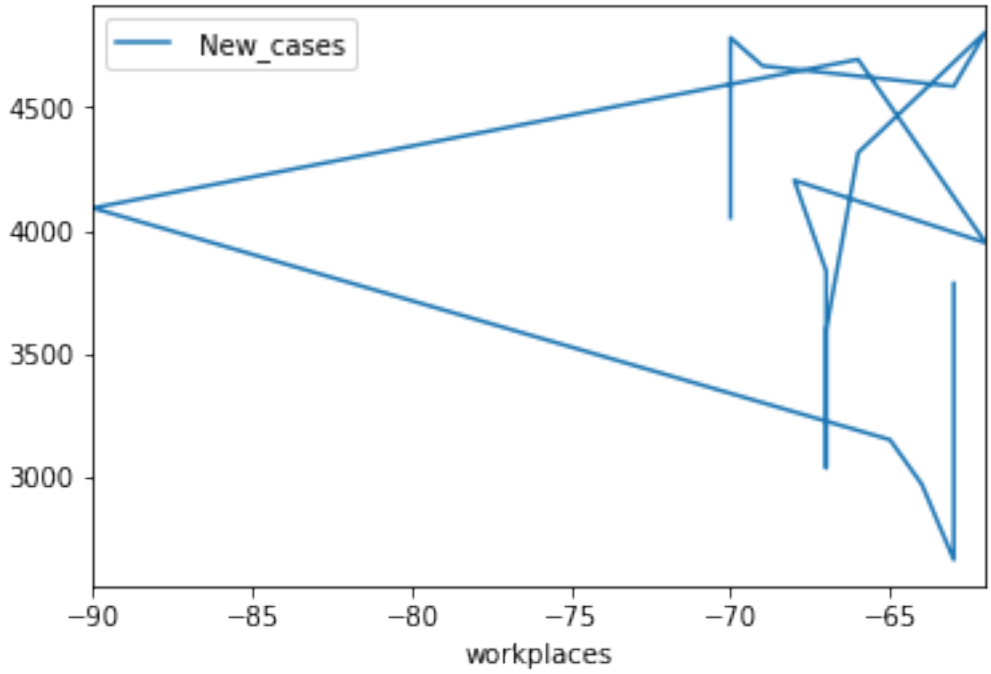




[38]:
```
italy_april_df.plot("date", "workplaces", title="Italy Workplace Mobility,␣
↪April")
italy_april_df.plot("workplaces", " New_cases")
```

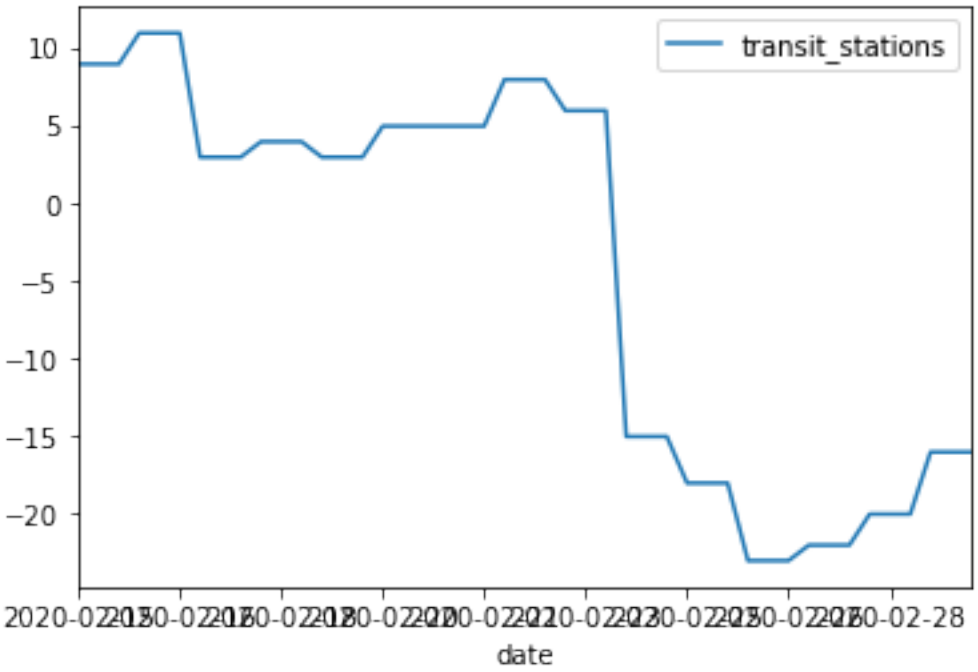
[38]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99db7b128>

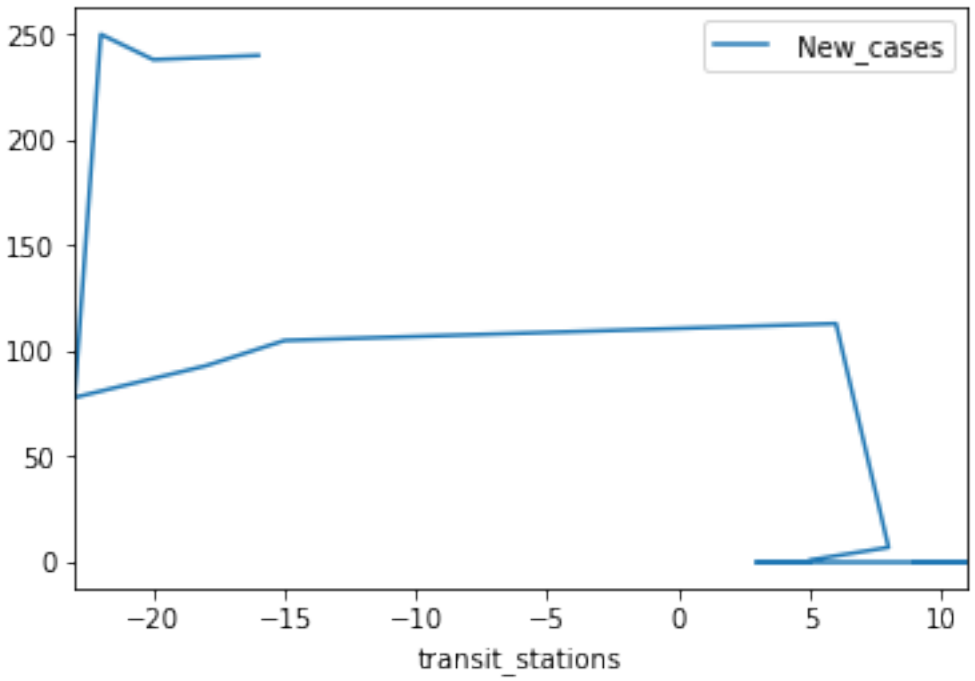

Italy Workplace Mobility, April
−65
−70
−75
−80
−85
−90
workplaces
2020-04-01
2020-04-04
2020-04-07
2020-04-11
2020-04-14
2020-04-17
date


New_cases
4500
4000
3500
3000
−90
−85
−80
−75
−70
−65
workplaces

[39]:
```
# Let's try with transit stations now.
italy_feb_df.plot("date", "transit_stations")
italy_feb_df.plot("transit_stations", " New_cases")
```

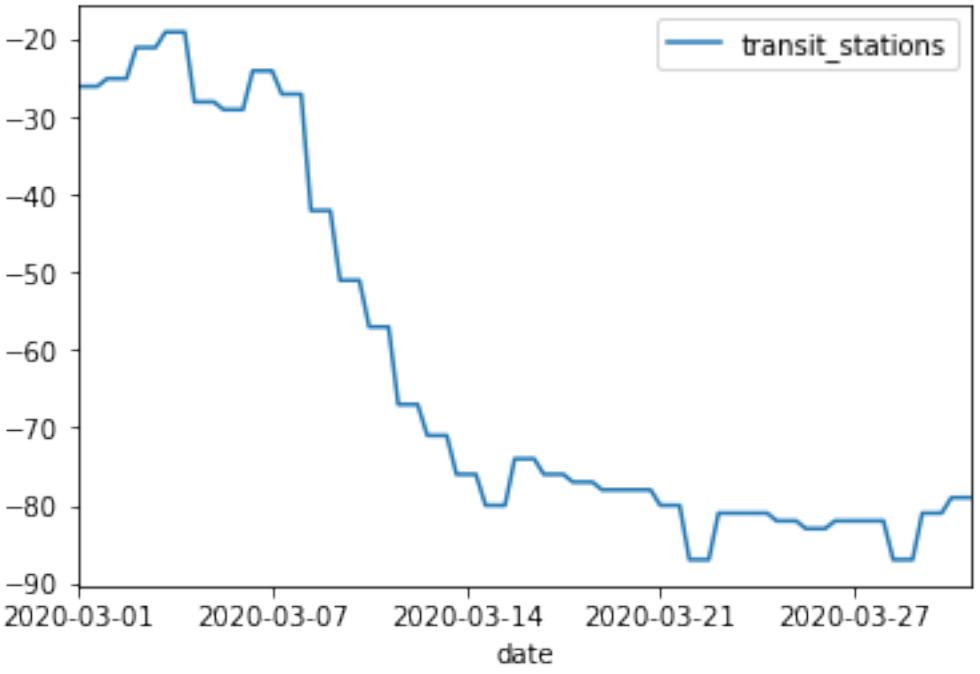
[39]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99db158d0>

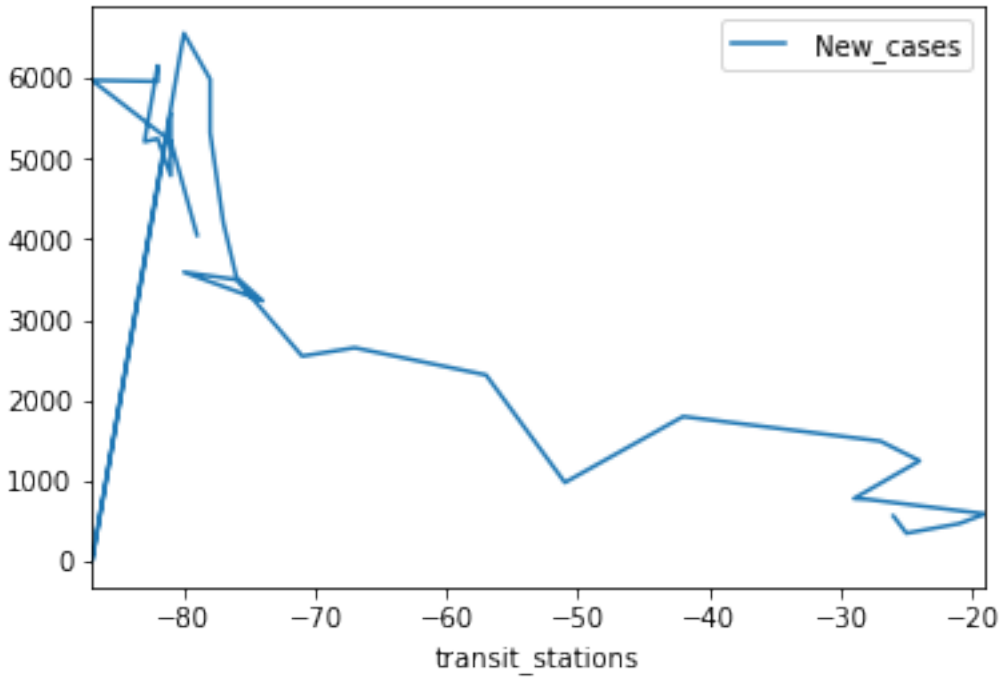




[40]:
```
italy_march_df.plot("date", "transit_stations")
italy_march_df.plot("transit_stations", " New_cases")
```

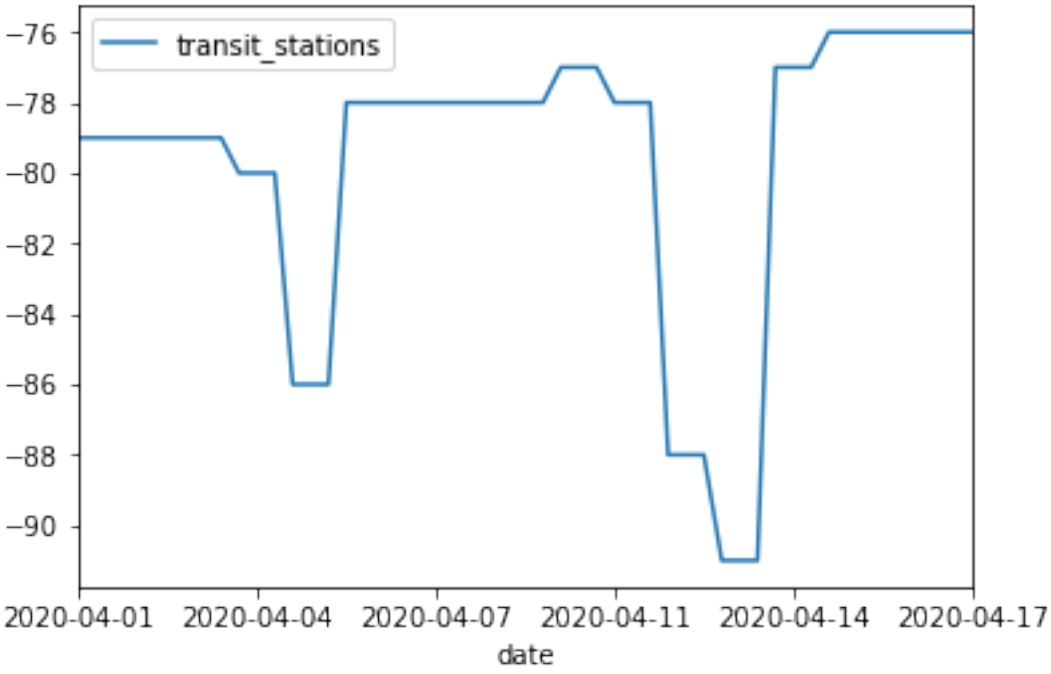
[40]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e05f198>

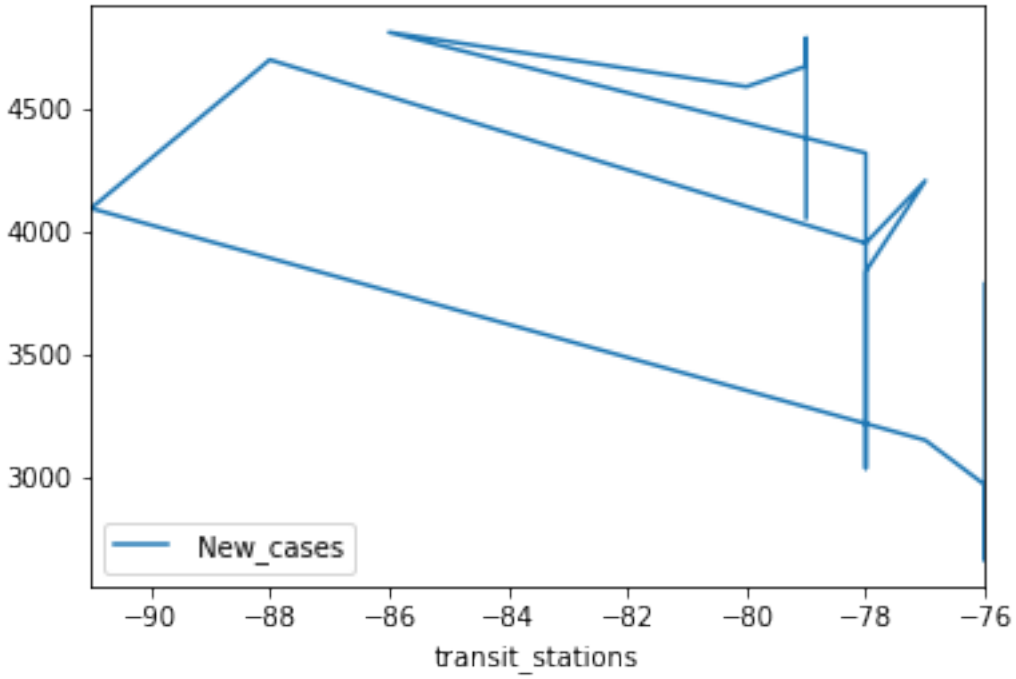




[41]:
```
italy_april_df.plot("date", "transit_stations")
italy_april_df.plot("transit_stations", " New_cases")
```

[41]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e115668>


transit_stations
−76
−78
−80
−82
−84
−86
−88
−90
2020-04-01
2020-04-04
2020-04-07
2020-04-11
2020-04-14
2020-04-17
date

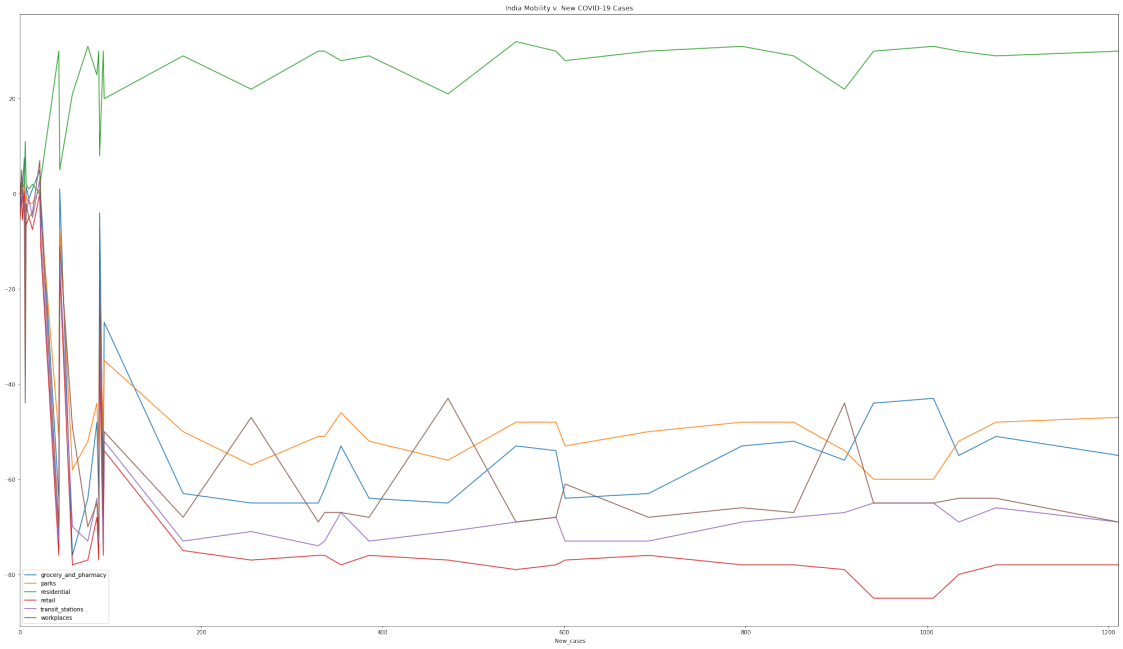

4500
4000
3500
3000
New_cases
−90
−88
−86
−84
−82
−80
−78
−76
transit_stations

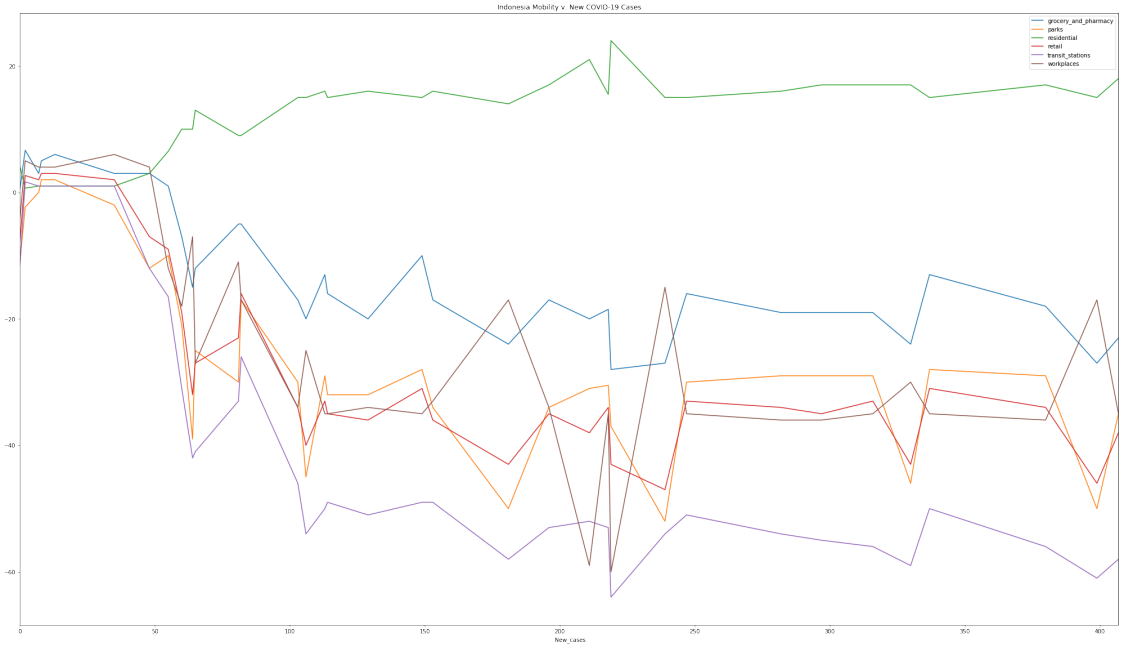
[42]:
```
# So far, I'm seeing the greatest correlation between mobility and new cases in␣
 ↪Italy during the month of March (makes sense to me)
```

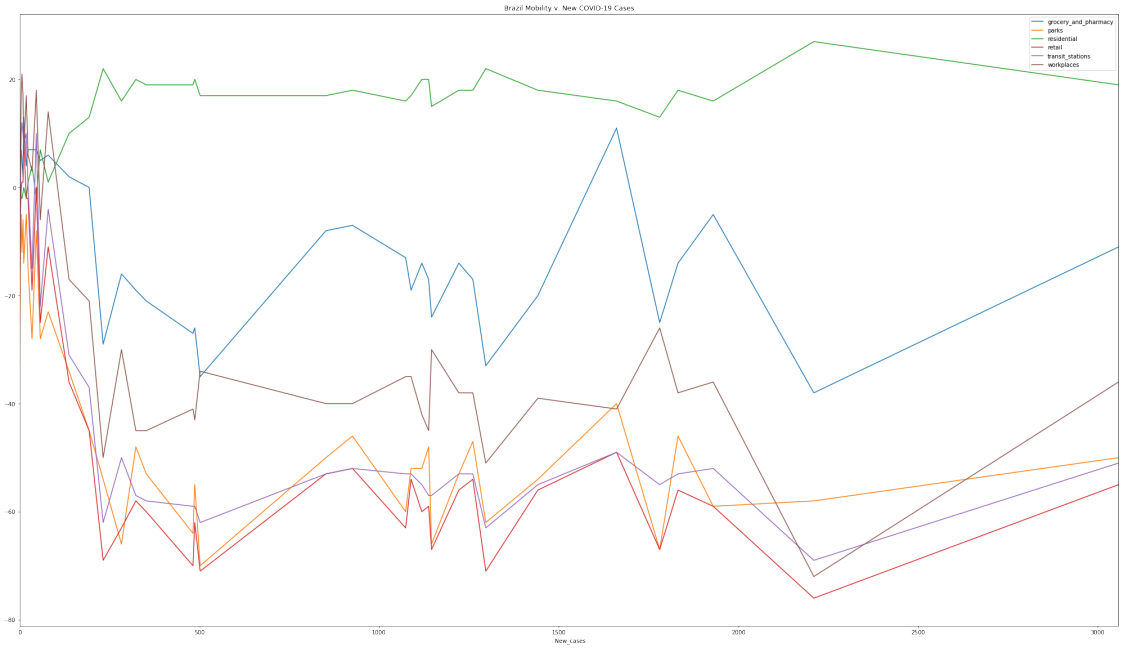
[43]:
```
# Let's see if we can find any countries where mobility corresponds more clearly␣
 ↪to new COVID infection rates
# We can check just the 10 largest countries in the dataset

india_df = mobility_df[mobility_df["country"].str.contains("India")]
# Let's format our dataframe the same way we did the earlier DF for all␣
 ↪countries combined, but this time using cumulative cases instead of time
india_df = india_df.sort_index(level=[" New_cases"])
india_df = india_df.groupby([' New_cases']).mean()
# Let's remove the "value" column and COVID case/death columns so that we can␣
 ↪just see the different modes of mobility
india_df = india_df[india_df.columns.difference(['value', " Cumulative_deaths",␣
 ↪" Cumulative_cases", " New_deaths"])]

# Now, we repeat this process for each of the remaining top-10 countries
indonesia_df = mobility_df[mobility_df["country"].str.contains("Indonesia")]
indonesia_df = indonesia_df.sort_index(level=[" New_cases"])
indonesia_df = indonesia_df.groupby([' New_cases']).mean()
indonesia_df = indonesia_df[indonesia_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

brazil_df = mobility_df[mobility_df["country"].str.contains("Brazil")]
brazil_df = brazil_df.sort_index(level=[" New_cases"])
brazil_df = brazil_df.groupby([' New_cases']).mean()
brazil_df = brazil_df[brazil_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

mexico_df = mobility_df[mobility_df["country"].str.contains("Mexico")]
mexico_df = mexico_df.sort_index(level=[" New_cases"])
mexico_df = mexico_df.groupby([' New_cases']).mean()
mexico_df = mexico_df[mexico_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

japan_df = mobility_df[mobility_df["country"].str.contains("Japan")]
japan_df = japan_df.sort_index(level=[" New_cases"])
japan_df = japan_df.groupby([' New_cases']).mean()
japan_df = japan_df[japan_df.columns.difference(['value', " Cumulative_deaths",␣
 ↪" Cumulative_cases", " New_deaths"])]

philippines_df = mobility_df[mobility_df["country"].str.contains("Philippines")]
philippines_df = philippines_df.sort_index(level=[" New_cases"])
philippines_df = philippines_df.groupby([' New_cases']).mean()
```

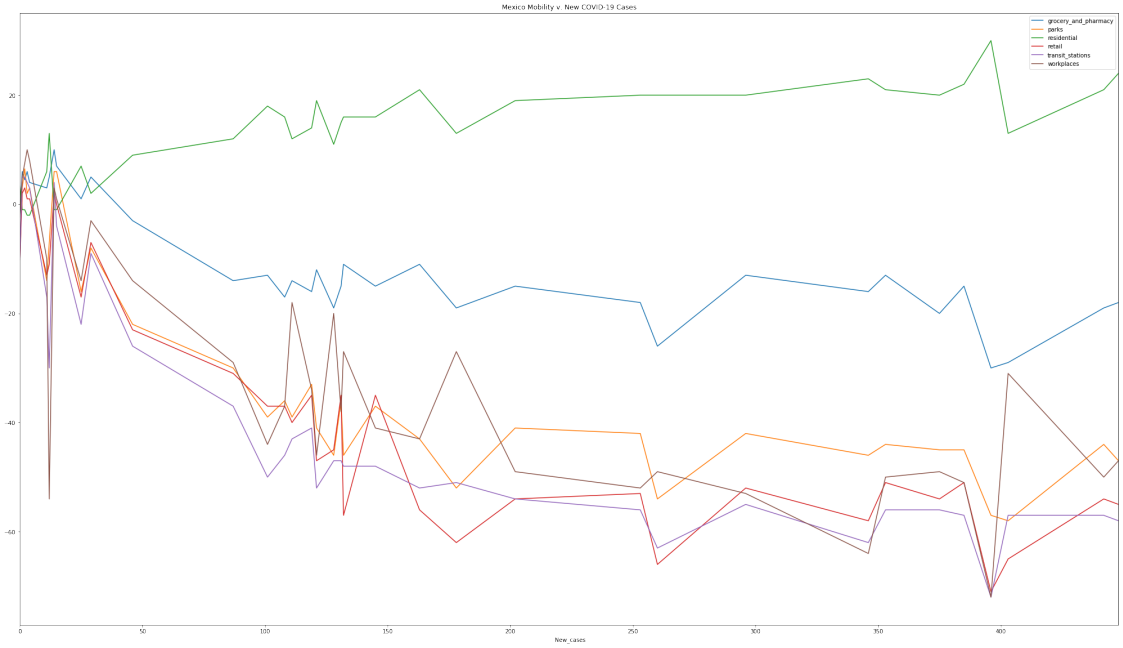
```
philippines_df = philippines_df[philippines_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

egypt_df = mobility_df[mobility_df["country"].str.contains("Egypt")]
egypt_df = egypt_df.sort_index(level=[" New_cases"])
egypt_df = egypt_df.groupby([' New_cases']).mean()
egypt_df = egypt_df[egypt_df.columns.difference(['value', " Cumulative_deaths",␣
 ↪" Cumulative_cases", " New_deaths"])]

turkey_df = mobility_df[mobility_df["country"].str.contains("Turkey")]
turkey_df = turkey_df.sort_index(level=[" New_cases"])
turkey_df = turkey_df.groupby([' New_cases']).mean()
turkey_df = turkey_df[turkey_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

germany_df = mobility_df[mobility_df["country"].str.contains("Germany")]
germany_df = germany_df.sort_index(level=[" New_cases"])
germany_df = germany_df.groupby([' New_cases']).mean()
germany_df = germany_df[germany_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

thailand_df = mobility_df[mobility_df["country"].str.contains("Thailand")]
thailand_df = thailand_df.sort_index(level=[" New_cases"])
thailand_df = thailand_df.groupby([' New_cases']).mean()
thailand_df = thailand_df[thailand_df.columns.difference(['value', "␣
 ↪Cumulative_deaths", " Cumulative_cases", " New_deaths"])]

# Finally, we can plot the data for these 10 countries
india_df.plot(figsize=(35,20), title="India Mobility v. New COVID-19 Cases")
indonesia_df.plot(figsize=(35,20), title="Indonesia Mobility v. New COVID-19␣
 ↪Cases")
brazil_df.plot(figsize=(35,20), title="Brazil Mobility v. New COVID-19 Cases")
mexico_df.plot(figsize=(35,20), title="Mexico Mobility v. New COVID-19 Cases")
japan_df.plot(figsize=(35,20), title="Japan Mobility v. New COVID-19 Cases")
philippines_df.plot(figsize=(35,20), title="The Philippines Mobility v. New␣
 ↪COVID-19 Cases")
egypt_df.plot(figsize=(35,20), title="Egypt Mobility v. New COVID-19 Cases")
turkey_df.plot(figsize=(35,20), title="Turkey Mobility v. New COVID-19 Cases")
germany_df.plot(figsize=(35,20), title="Germany Mobility v. New COVID-19 Cases")
thailand_df.plot(figsize=(35,20), title="Thailand Mobility v. New COVID-19␣
 ↪Cases")
```

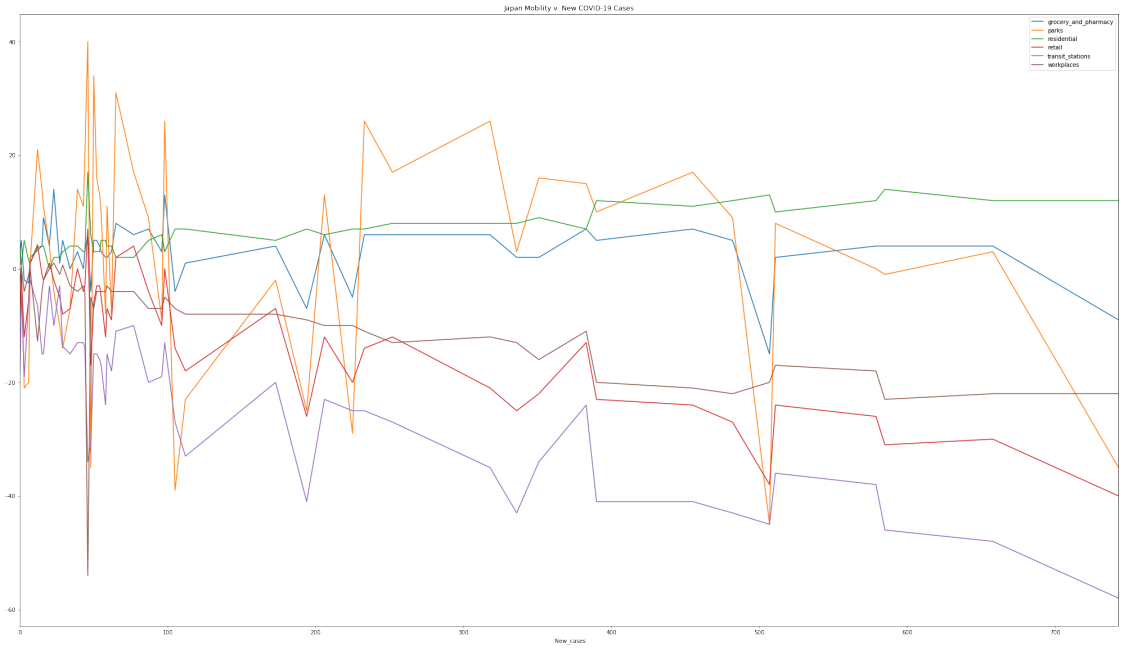
[43]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e586fd0>

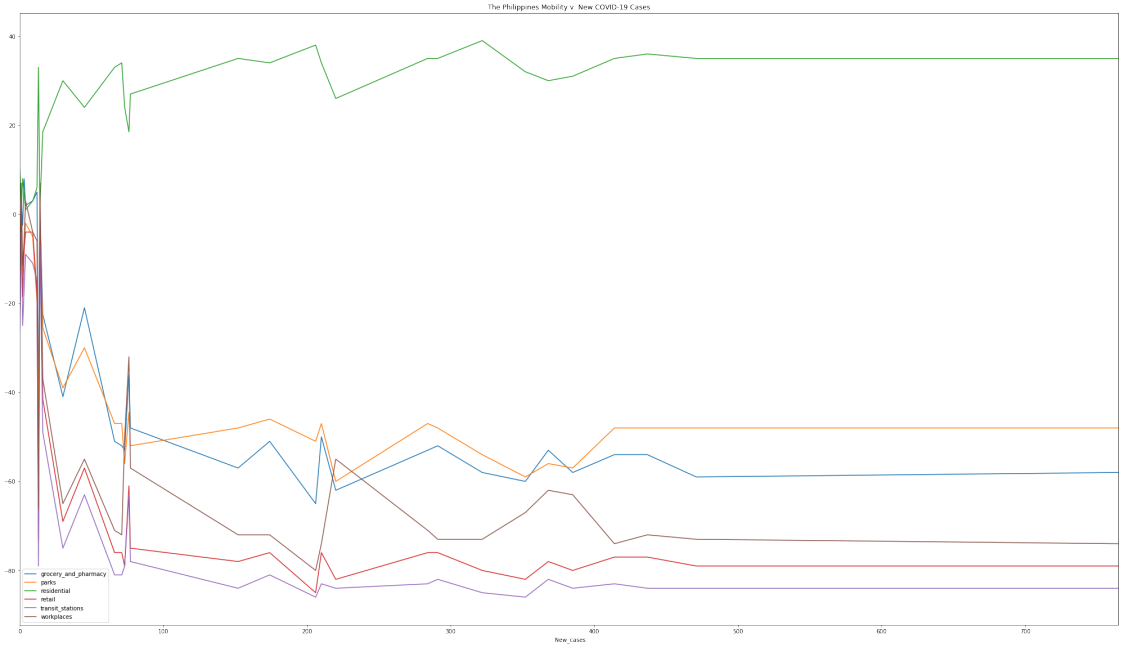

India Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
New_cases

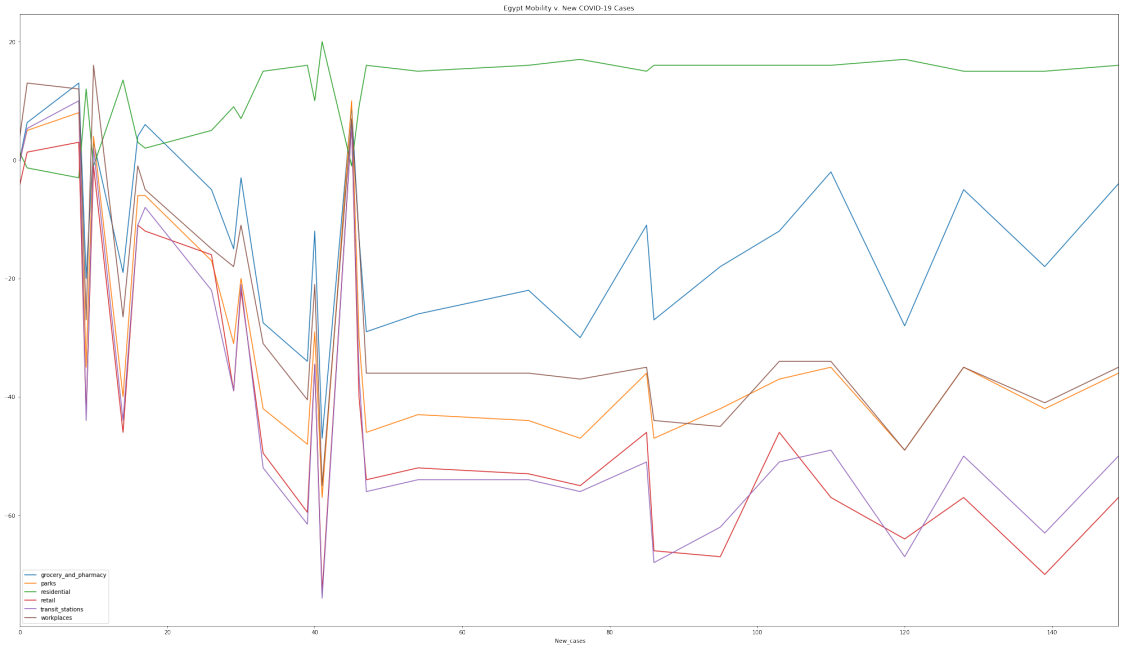

Indonesia Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
New_cases

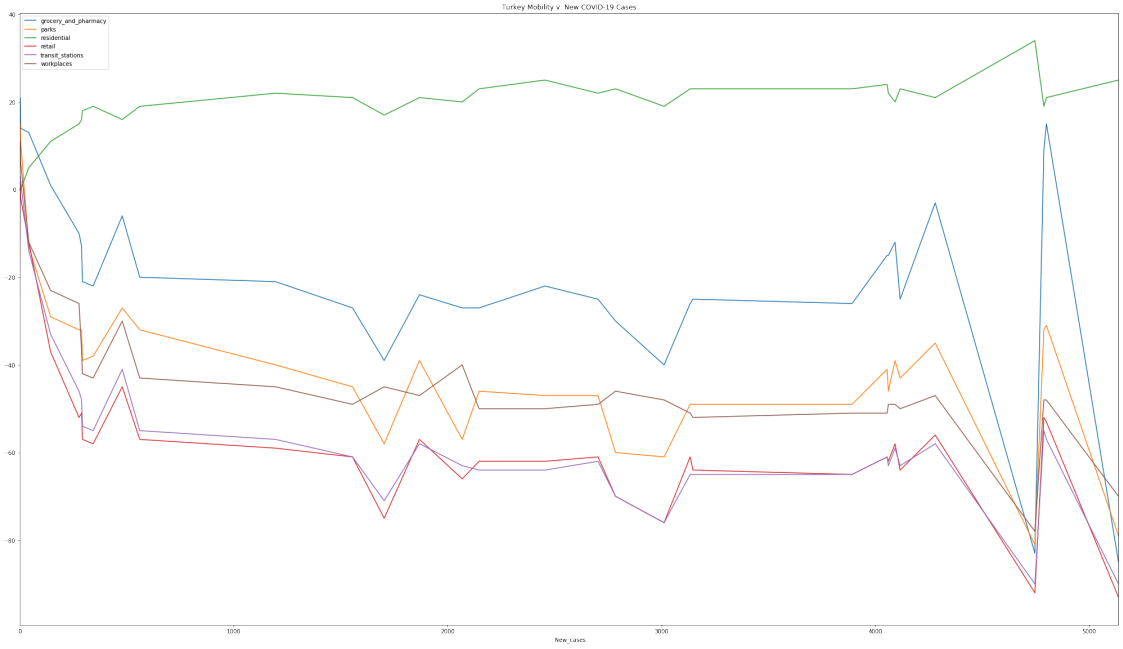

Brazil Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
20
0
-20
-40
-60
-80
0
500
1000
1500
2000
2500
3000
New_cases

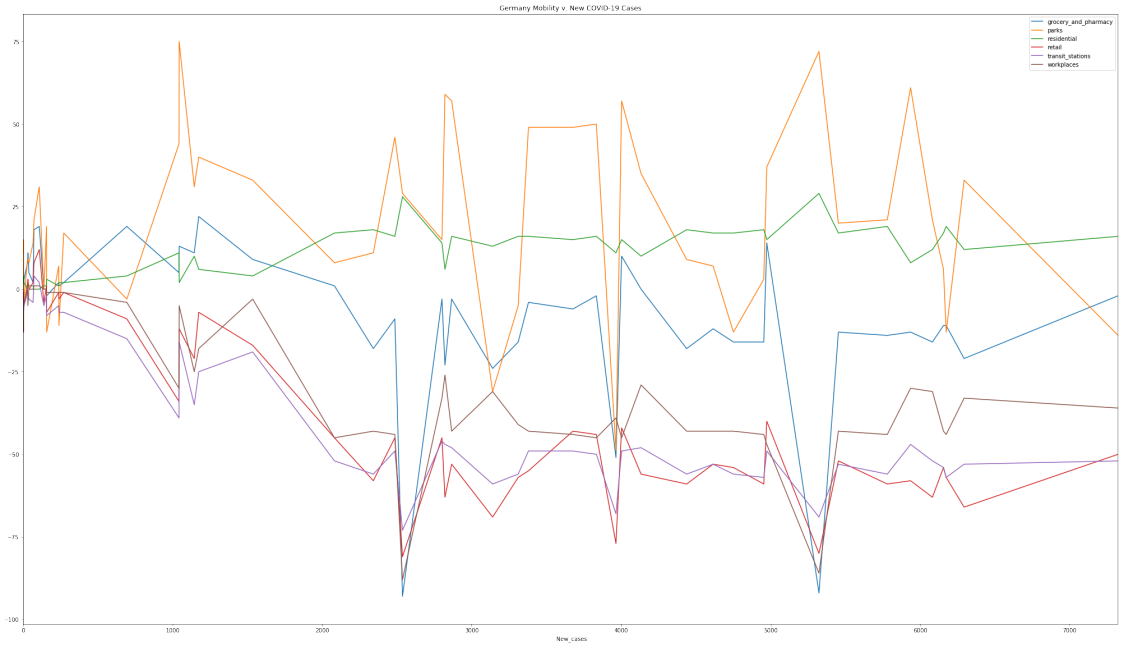

Mexico Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
20
0
-20
-40
-60
0
50
100
150
200
250
300
350
400
New_cases

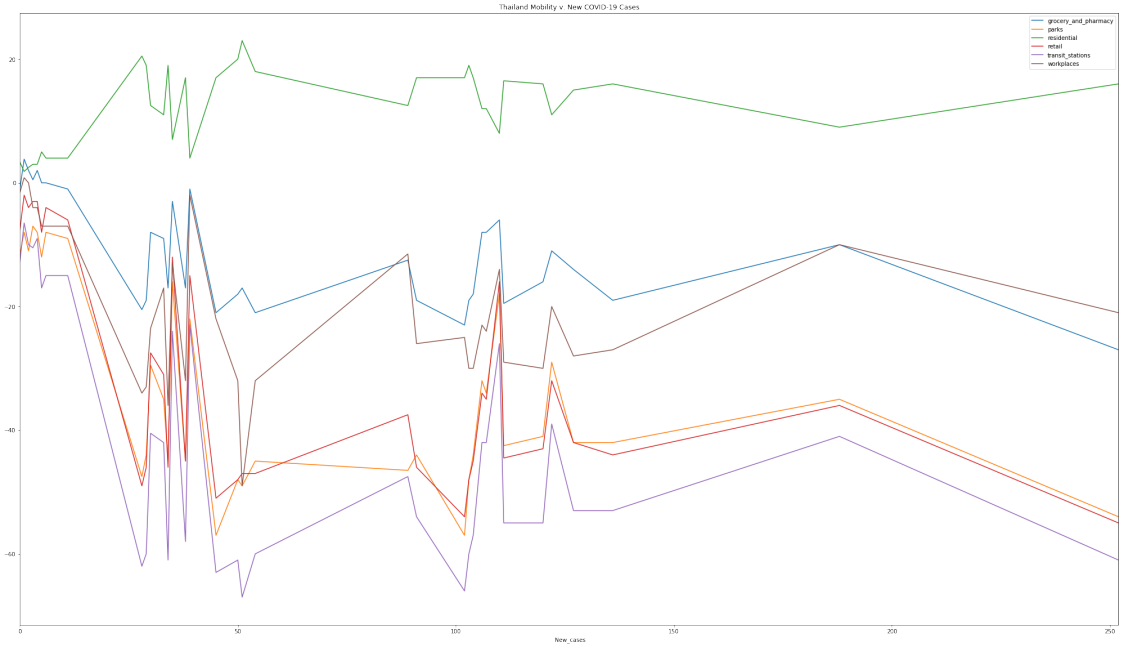

Japan Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
40
20
0
-20
-40
-60
0
100
200
300
400
500
600
700
New_cases


The Philippines Mobility v. New COVID-19 Cases
40
20
0
-20
-40
-60
-80
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
0
100
200
300
400
500
600
700
New_cases


Egypt Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
New_cases

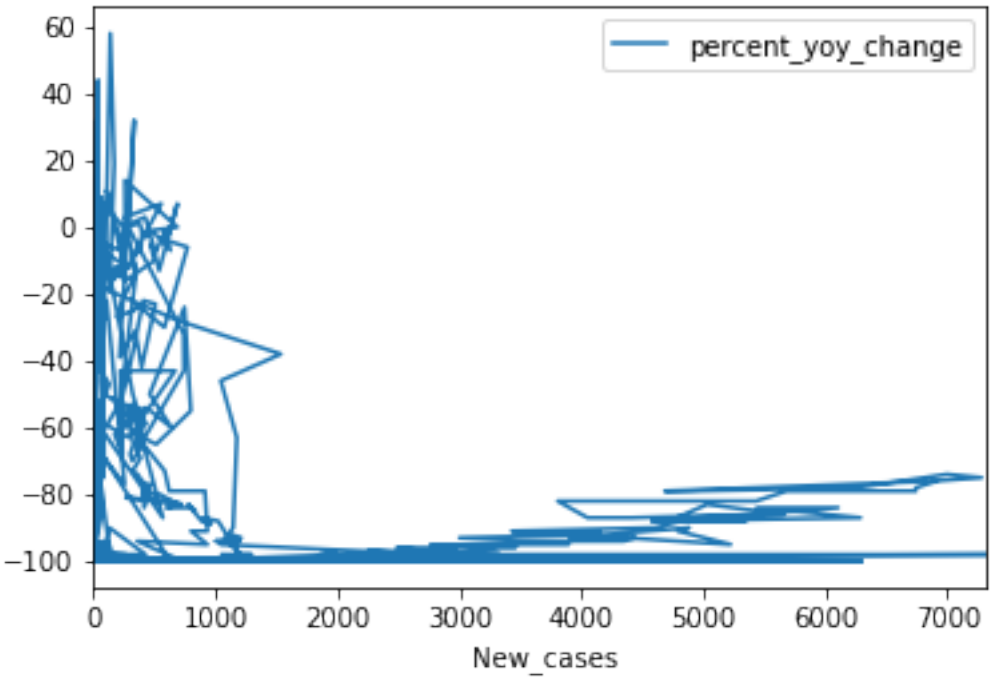

Turkey Mobility v. New COVID-19 Cases
grocery_and_pharmacy
parks
residential
retail
transit_stations
workplaces
New_cases





[44]:
```
# Let's start out plotting correlation between restaurant performance and new␣
 ↪cases of COVID-19
# Note that this is on a global scale (we can break it up by country later)
restaurant_df.plot(" New_cases", "percent_yoy_change")
```

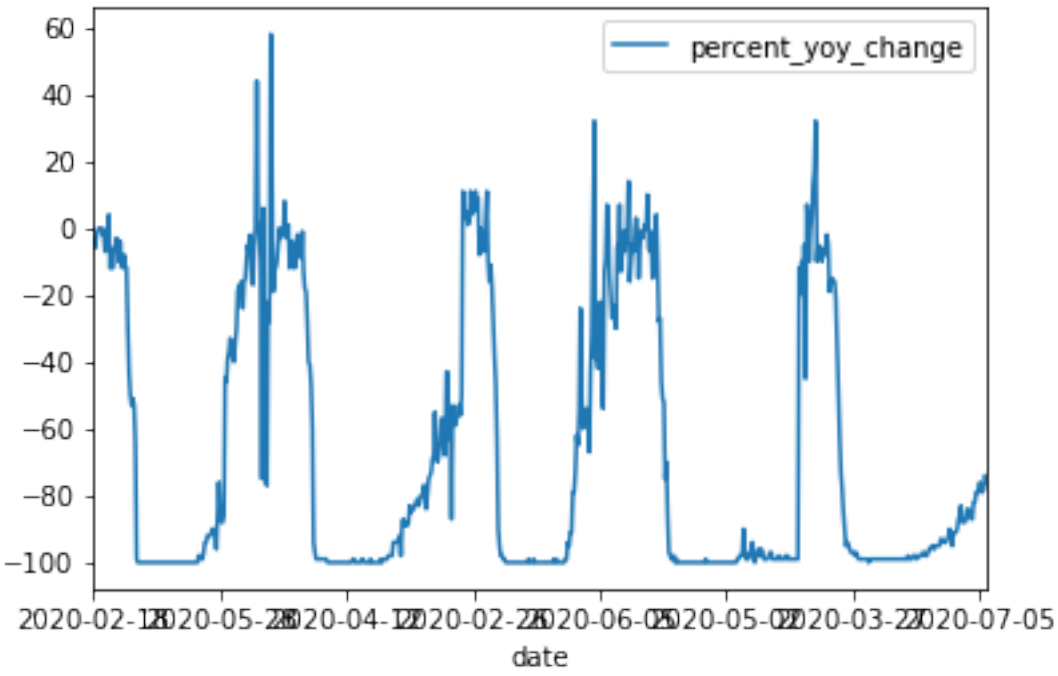
[44]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e6b4400>



Looks like restaurant performance was generally higher when the number of new cases in a given country was below 1,000. However, this could be due to mandatory restaurant closures at a certain date.

[45]:
```
# Let's see how restaurant performace changed over time instead
restaurant_df.plot("date", "percent_yoy_change")
```

[45]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e744ba8>

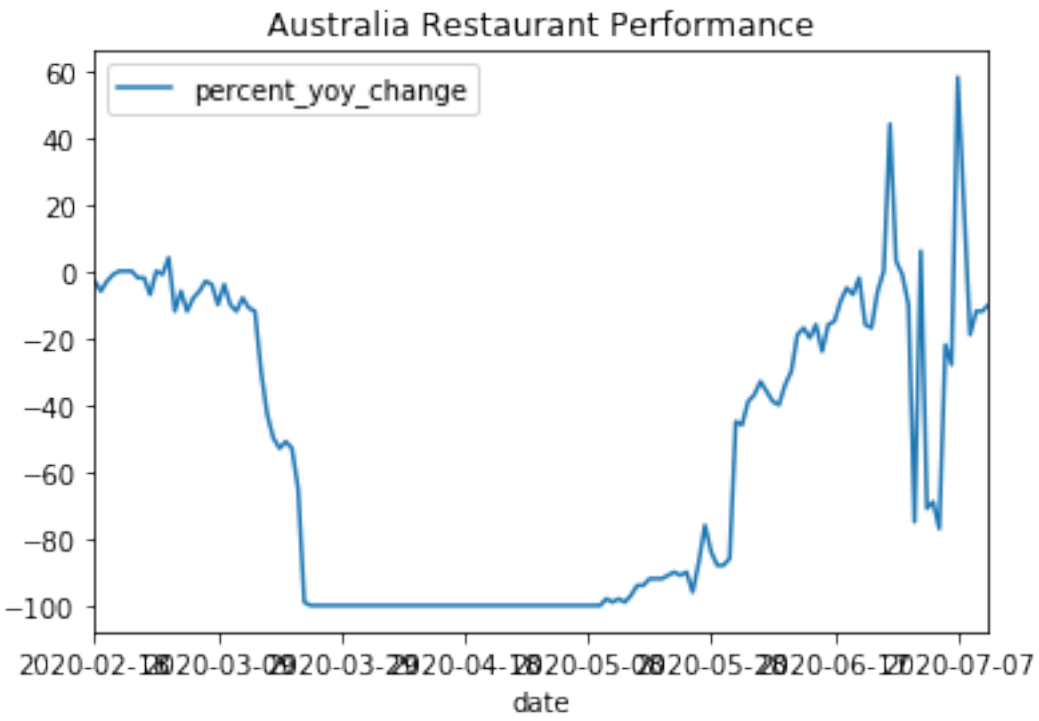


Now we get 5 spikes. I wonder if each spike reflects restaurant closures in each of the 5 countries included in the dataset? Let's look at the countries individually.

[46]:
```
# Let's break this up into 5 dataframes, one for each country
aus_df = restaurant_df[restaurant_df["country"].str.contains("Australia")]
can_df = restaurant_df[restaurant_df["country"].str.contains("Canada")]
ger_df = restaurant_df[restaurant_df["country"].str.contains("Germany")]
ire_df = restaurant_df[restaurant_df["country"].str.contains("Ireland")]
mex_df = restaurant_df[restaurant_df["country"].str.contains("Mexico")]
```

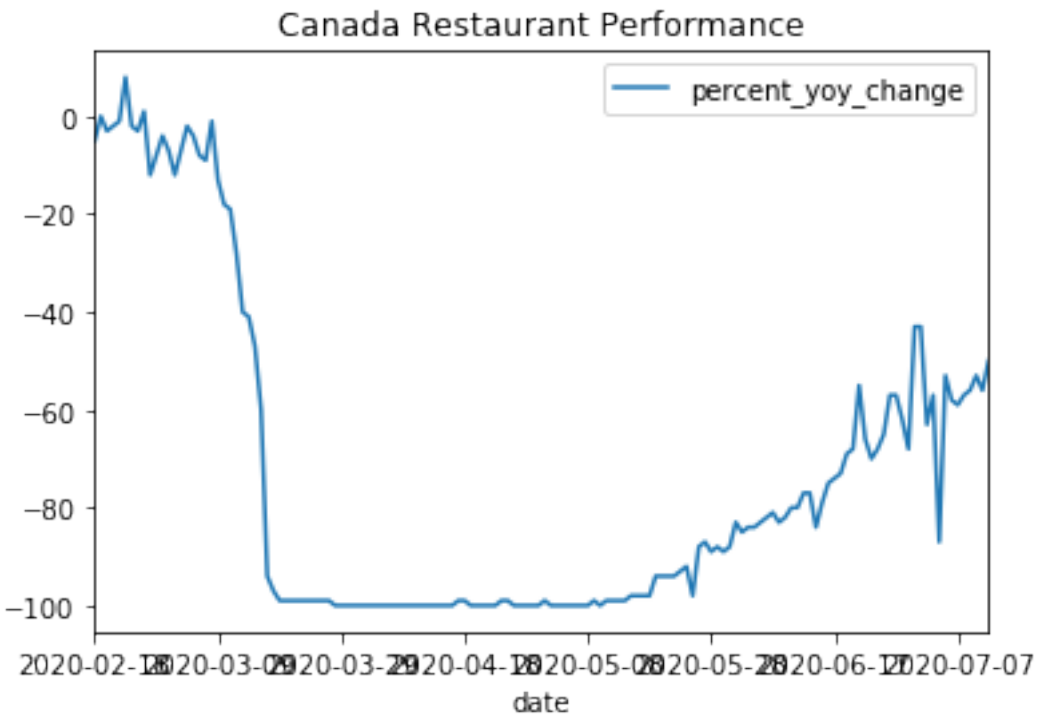
[47]:
```
# Now, let's see how restaurant performance changed over time in each individual␣
↪country

aus_df.plot("date", "percent_yoy_change", title="Australia Restaurant␣
↪Performance")
```

[47]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a99e7b0da0>
```



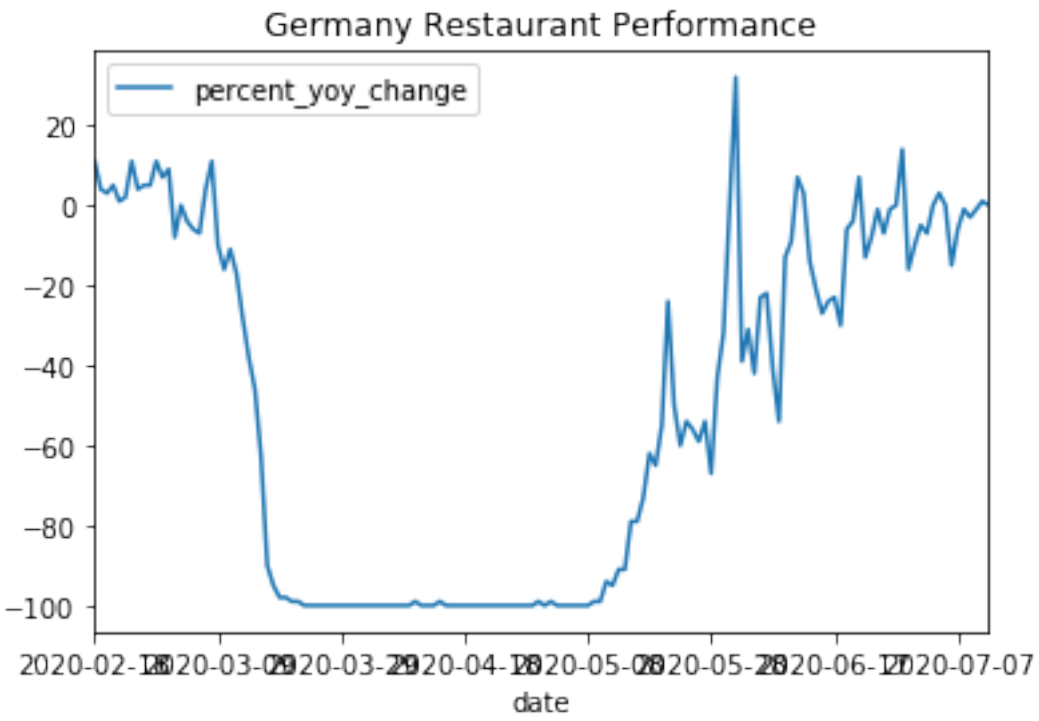
[48]:
```
can_df.plot("date", "percent_yoy_change", title="Canada Restaurant Performance")
```

[48]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a99e82f390>
```



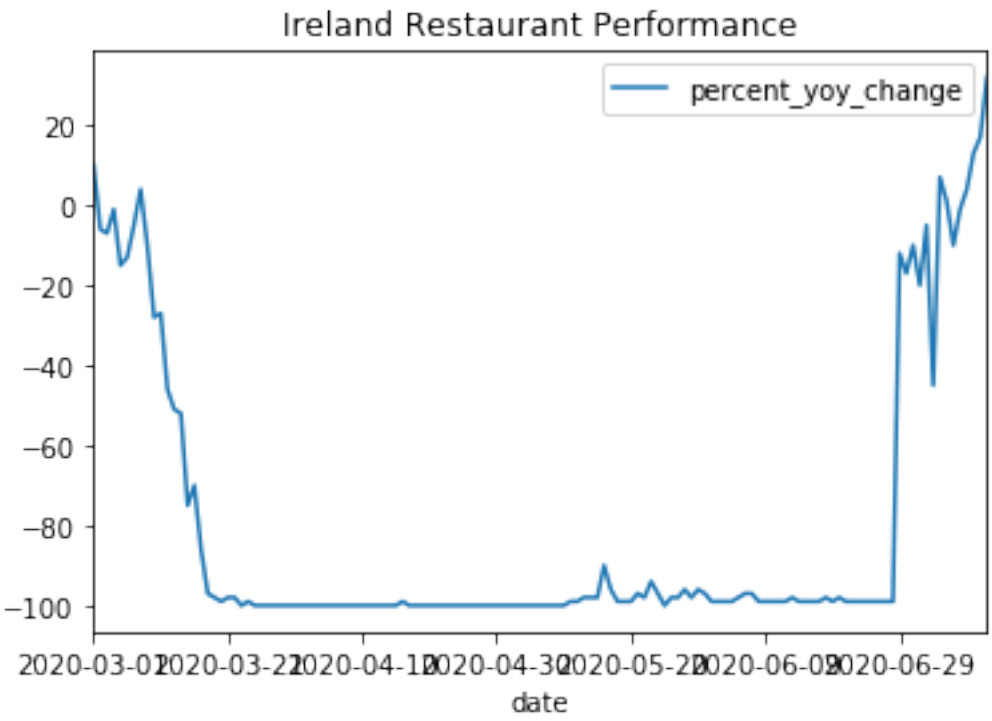
[49]:
```
ger_df.plot("date", "percent_yoy_change", title="Germany Restaurant Performance")
```

[49]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e53f2b0>



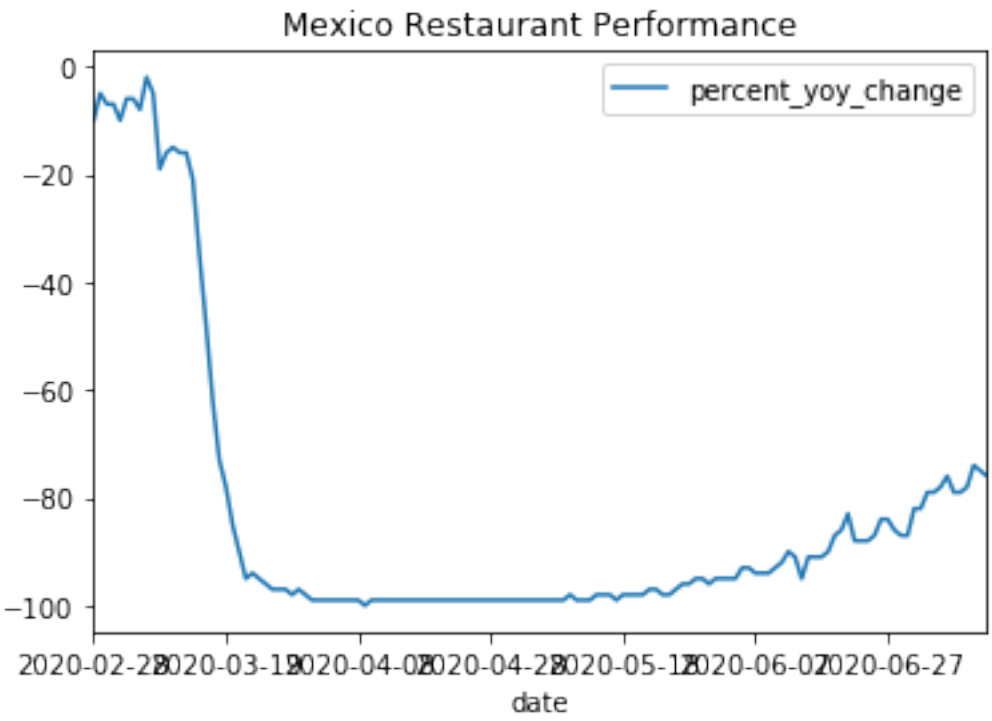
[50]:
```
ire_df.plot("date", "percent_yoy_change", title="Ireland Restaurant Performance")
```

[50]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a99e5301d0>
```



[51]:
```
mex_df.plot("date", "percent_yoy_change", title="Mexico Restaurant Performance")
```

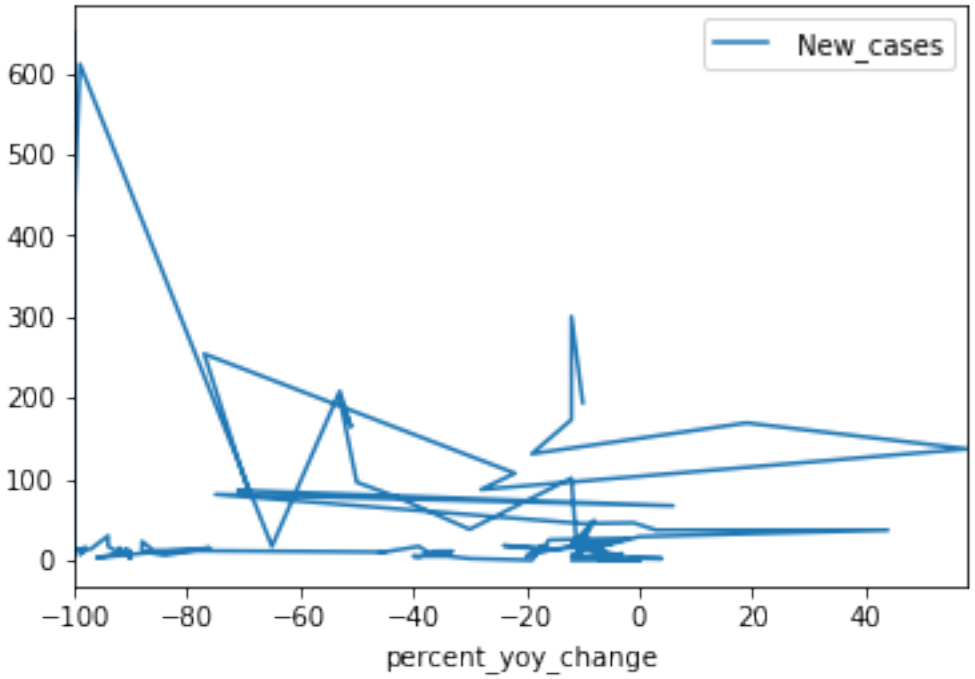
[51]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e3f4400>



In these plots, we can see high performance on earlier dates, followed by a steep decline (presumably due to mandatory restaurant closures), a low plateau, and then an increase at later dates (possibly due to restaurant reopenings, or maybe increasing business through alternative avenues like takeout).

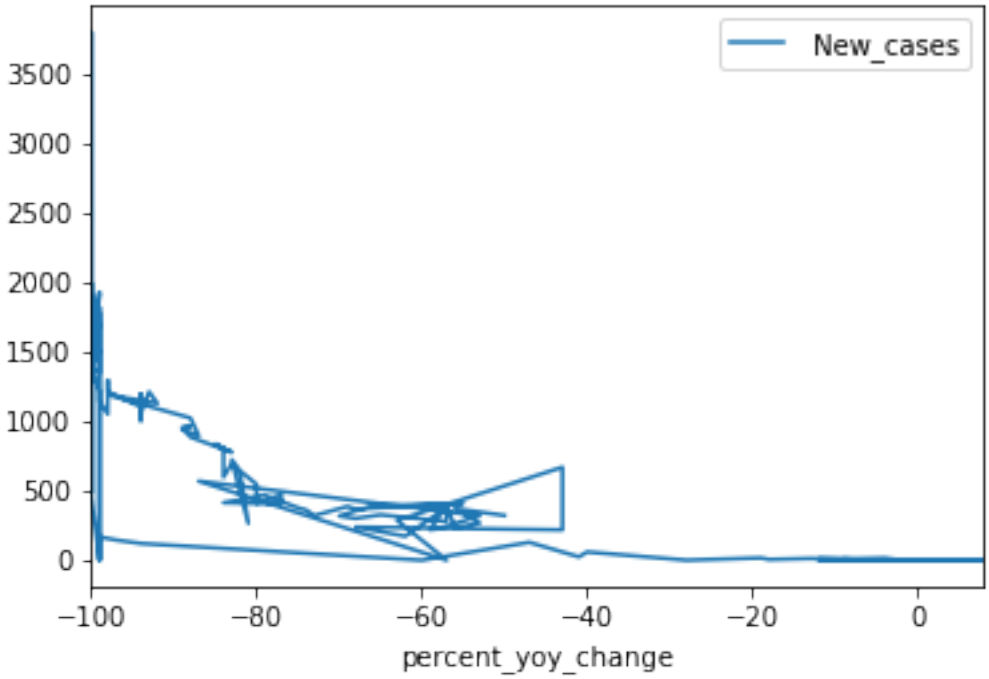
[52]:
```
# Now, let's see if these dips in restaurant performance (possibly due to␣
 ↪mandatory closures) had any correlation with the number of new COVID cases in␣
 ↪each individual country

aus_df.plot("percent_yoy_change", " New_cases")
```

[52]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1a99e37c390>
```



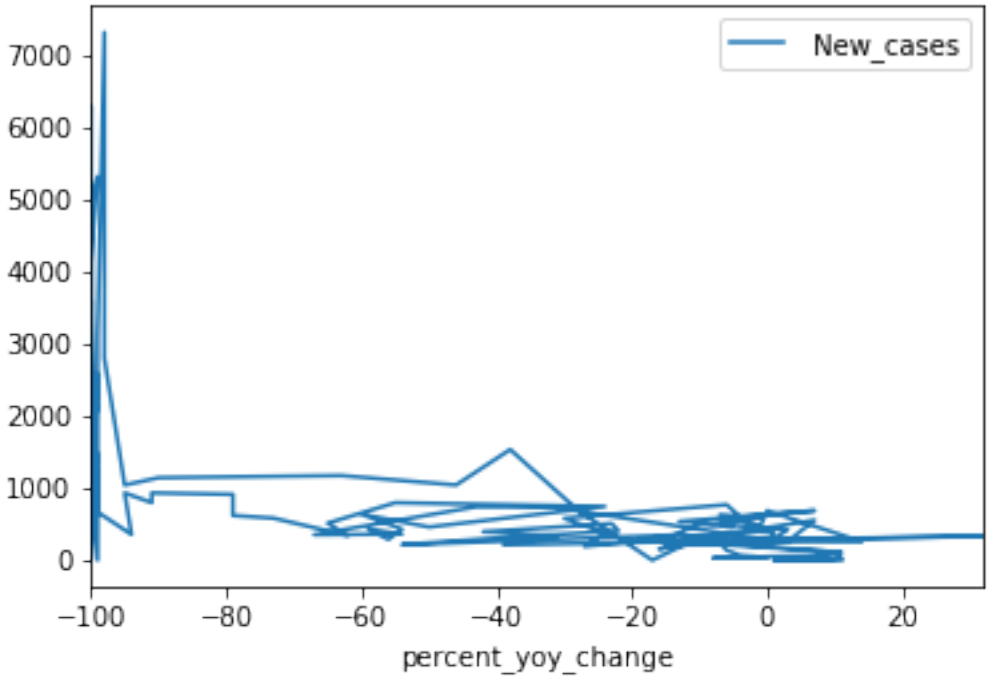
[53]:
```
can_df.plot("percent_yoy_change", " New_cases")
```

[53]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e1952e8>



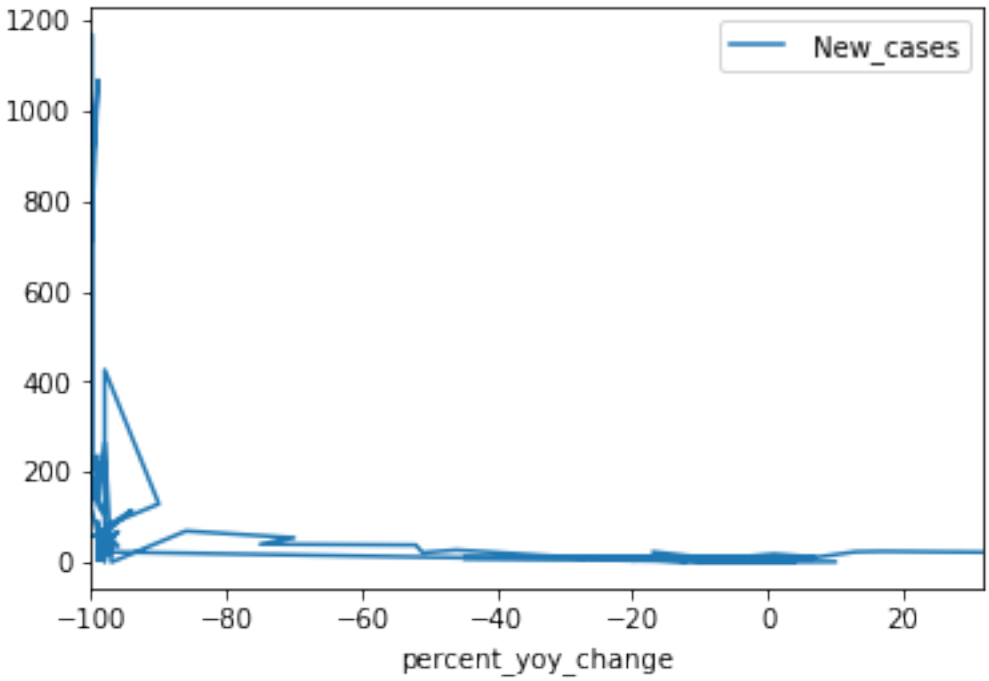
[54]:
```
ger_df.plot("percent_yoy_change", " New_cases")
```

[54]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99dabfa20>



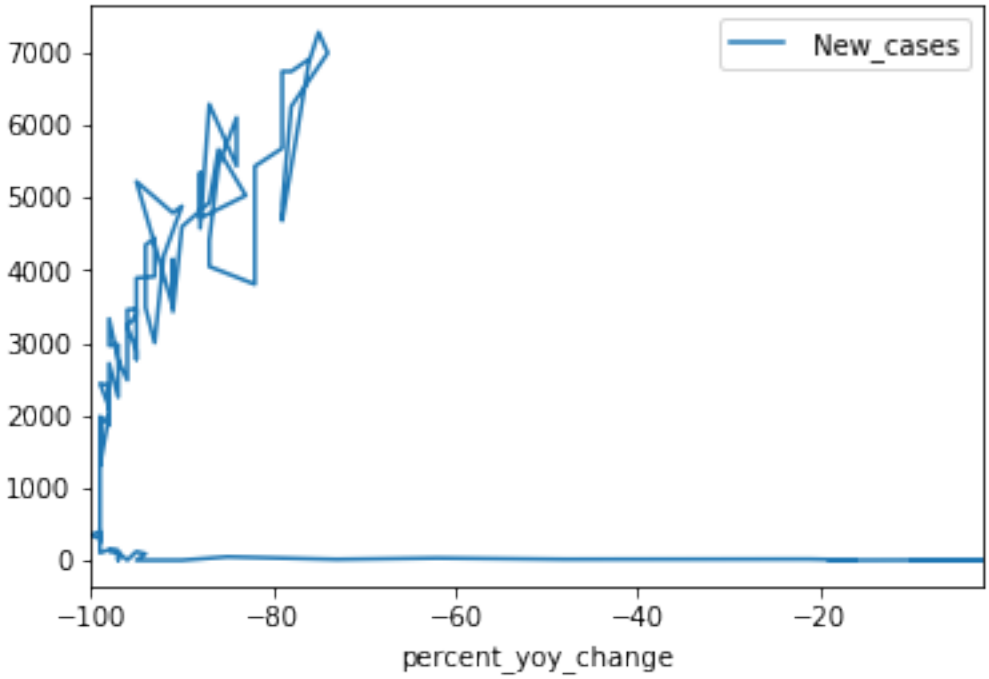
[55]:
```
ire_df.plot("percent_yoy_change", " New_cases")
```

[55]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e3d2080>



[56]:
```
mex_df.plot("percent_yoy_change", " New_cases")
```

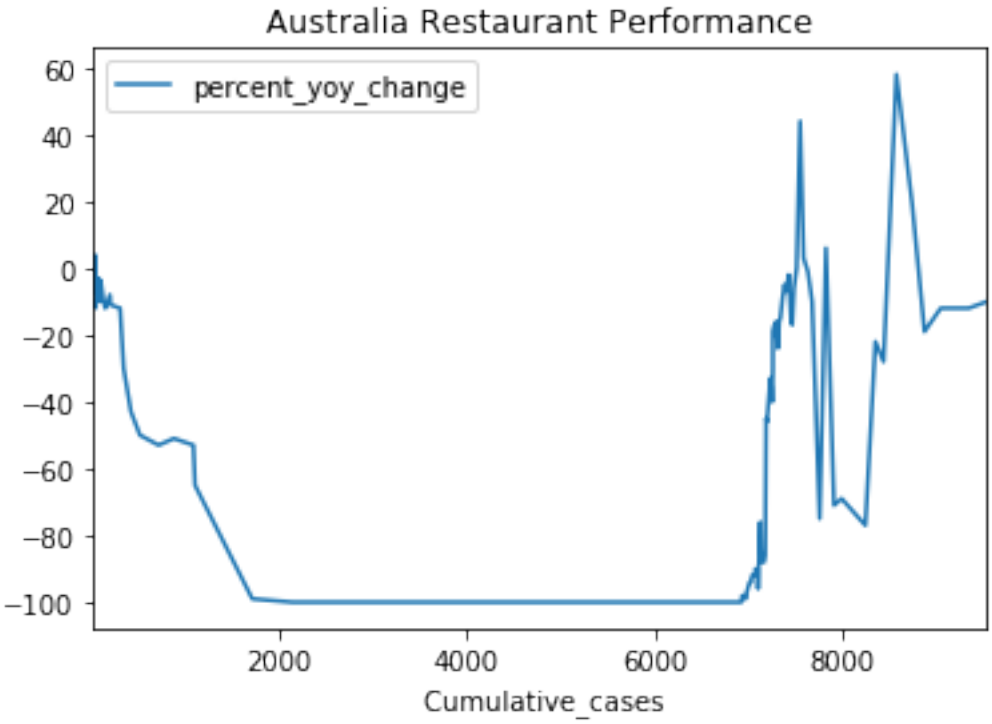
[56]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e4e86d8>



Interestingly, the number of new cases seems to decrease as restaurant business increases. This is most likely due to the fact that restaurants closed when new case rates were high, and then slowly reopened as new case numbers began to decrease. Let's try looking at total cases instead.

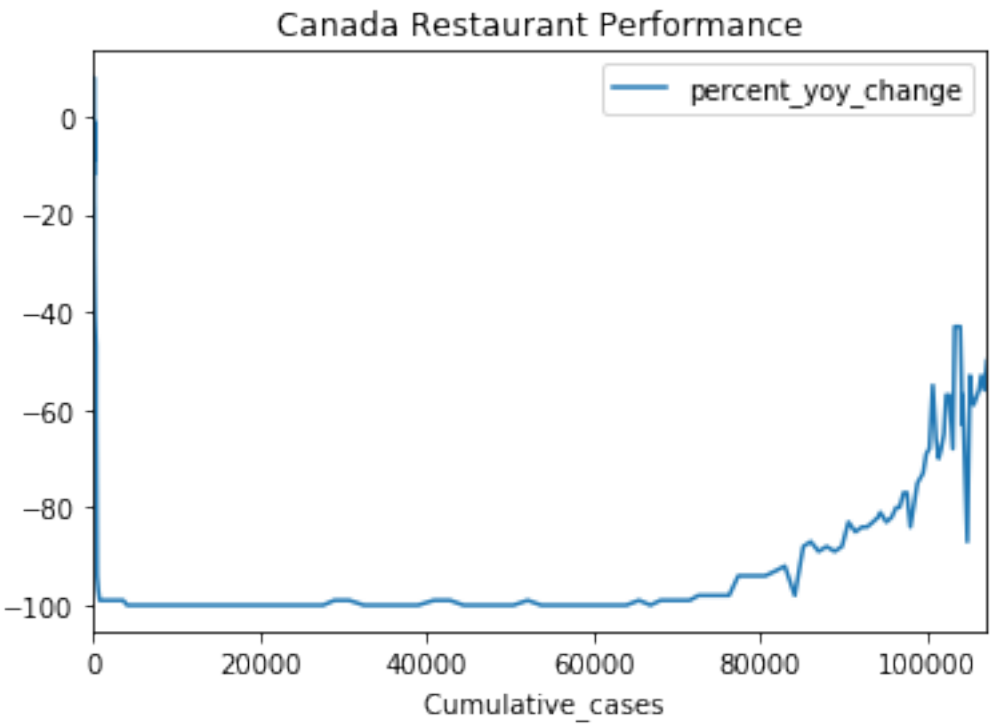
[57]:
```
aus_df.plot(" Cumulative_cases", "percent_yoy_change", title="Australia␣
↪Restaurant Performance")
```

[57]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e3821d0>



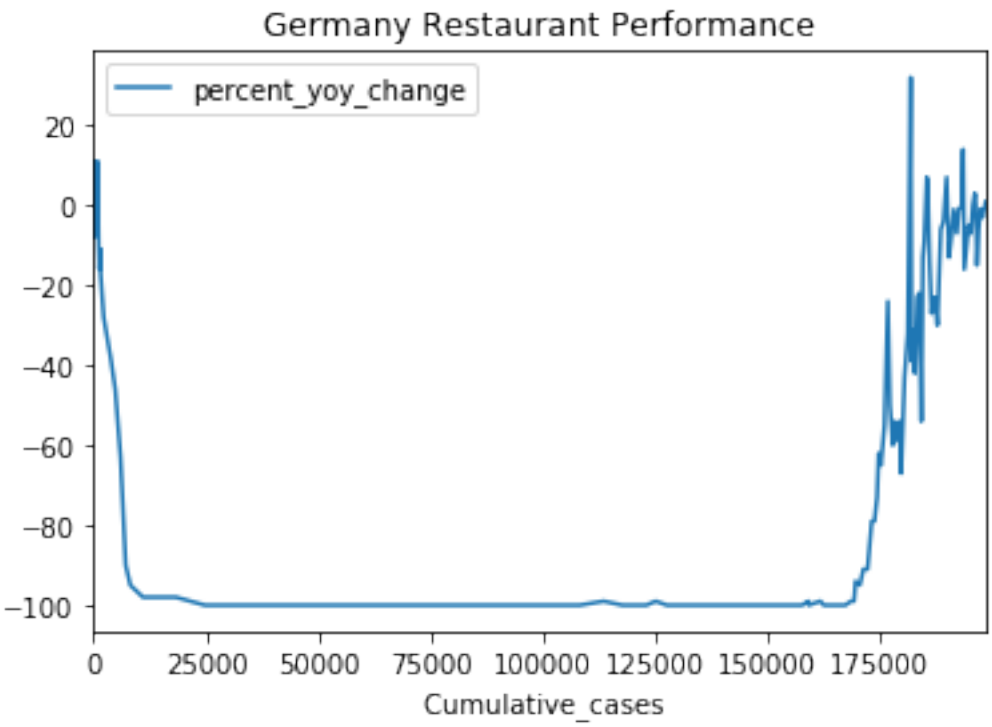
[58]:
```
can_df.plot(" Cumulative_cases", "percent_yoy_change", title="Canada Restaurant␣
↪Performance")
```

[58]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e243710>



[59]:
```
ger_df.plot(" Cumulative_cases", "percent_yoy_change", title="Germany Restaurant␣
↪Performance")
```

[59]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e326128>

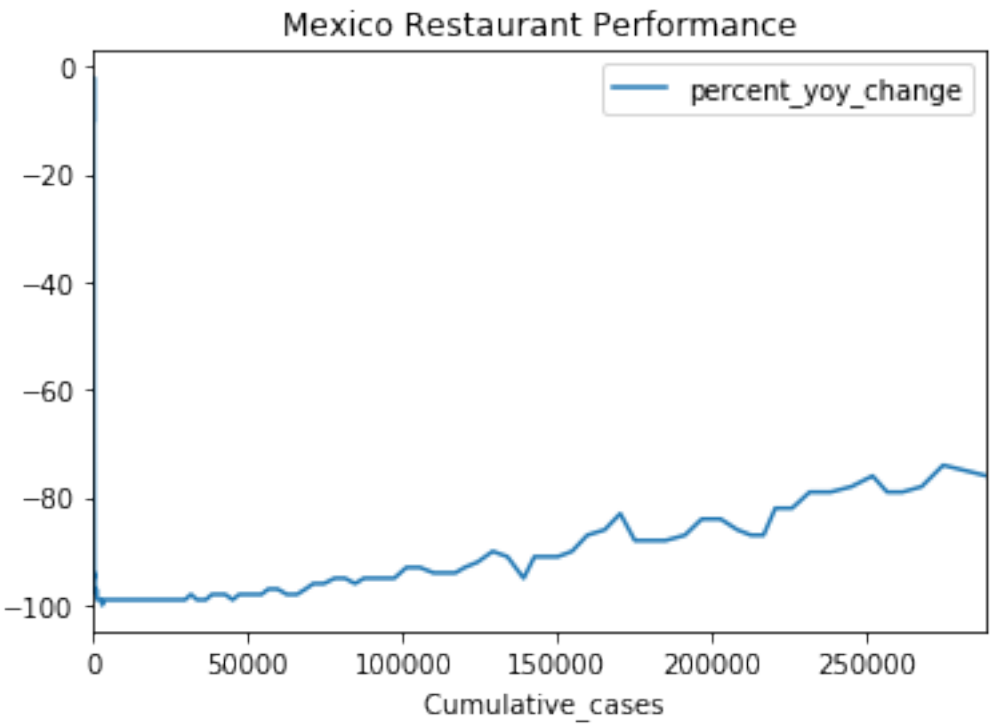


[60]:
```
ire_df.plot(" Cumulative_cases", "percent_yoy_change", title="Ireland Restaurant␣
↪Performance")
```

[60]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e216080>

[61]:
```
mex_df.plot(" Cumulative_cases", "percent_yoy_change", title="Mexico Restaurant␣
↪Performance")
```

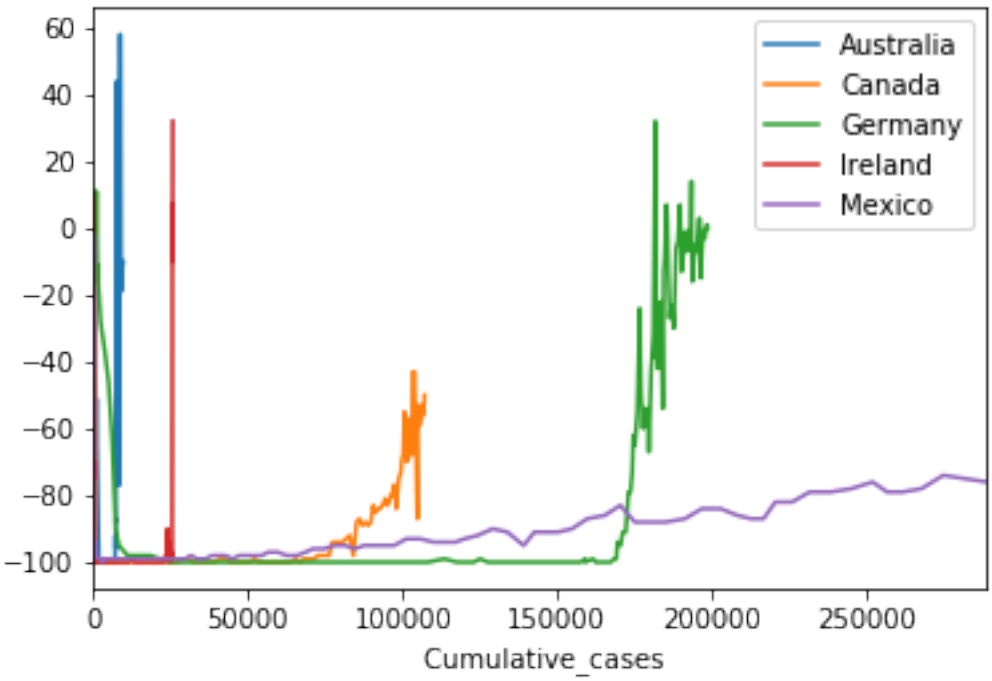
[61]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e468f28>



The above plots all have similar shapes. This is unsurprising, as restaurant business was best when cumulative cases were very low (before the worst of the pandemic and mandatory closures) and when cumulative cases were at their highest (towards the end of the worst of the pandemic, when restaurants were allowed to reopen).

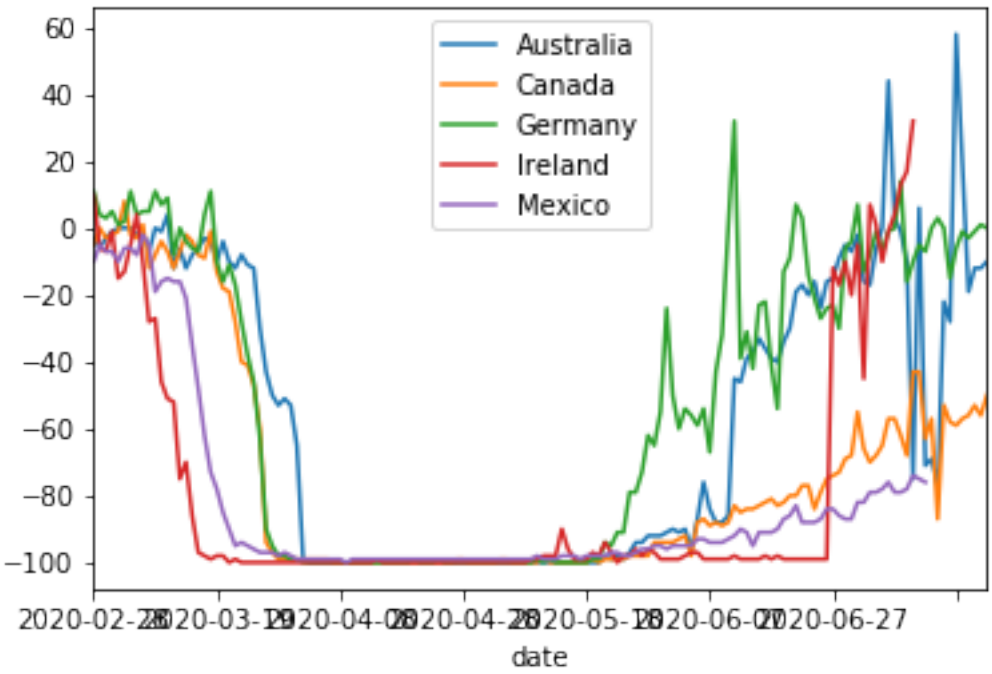
[62]:
```
# Let's compare individual countries with respect to cumulative cases and␣
 ↪restaurant performance change
ax = aus_df.plot(" Cumulative_cases", "percent_yoy_change")
ax = can_df.plot(" Cumulative_cases", "percent_yoy_change", ax=ax)
ax = ger_df.plot(" Cumulative_cases", "percent_yoy_change", ax=ax)
ax = ire_df.plot(" Cumulative_cases", "percent_yoy_change", ax=ax)
ax = mex_df.plot(" Cumulative_cases", "percent_yoy_change", ax=ax)

ax.legend(['Australia', 'Canada', 'Germany', 'Ireland', 'Mexico'])
ax.plot(figsize=(60,40))
```

[62]: []



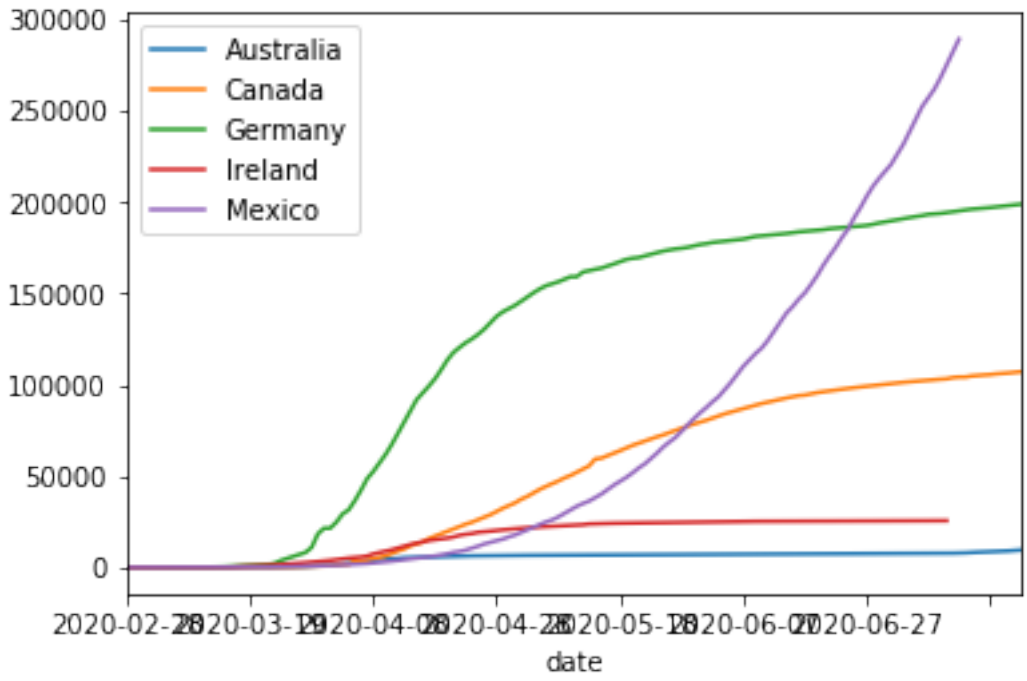
[63]:
```
# And now a time series comparison between the countries
ax = aus_df.plot("date", "percent_yoy_change")
ax = can_df.plot("date", "percent_yoy_change", ax=ax)
ax = ger_df.plot("date", "percent_yoy_change", ax=ax)
ax = ire_df.plot("date", "percent_yoy_change", ax=ax)
ax = mex_df.plot("date", "percent_yoy_change", ax=ax)

ax.legend(['Australia', 'Canada', 'Germany', 'Ireland', 'Mexico'])
ax.plot(figsize=(60,40))
```

[63]: []



[64]:
```
# Now, let's compare cumulative cases across the countries over time
ax = aus_df.plot("date", " Cumulative_cases")
ax = can_df.plot("date", " Cumulative_cases", ax=ax)
ax = ger_df.plot("date", " Cumulative_cases", ax=ax)
ax = ire_df.plot("date", " Cumulative_cases", ax=ax)
ax = mex_df.plot("date", " Cumulative_cases", ax=ax)

ax.legend(['Australia', 'Canada', 'Germany', 'Ireland', 'Mexico'])
ax.plot(figsize=(60,40))
```

[64]: []



2 Mobility/Restaurant Data (Domestic)

[65]:
```
%%javascript
IPython.OutputArea.prototype._should_scroll = function(lines) {
    return false;
}
```

```
<IPython.core.display.Javascript object>
```

[66]:
```
plt.rcParams.update({'figure.max_open_warning': 0})
```

[67]:
```
cases = pd.read_csv("data/us-states.csv")
cases['date'] = cases['date'].astype('datetime64[ns]')
cases = cases.drop(["fips"], axis=1)
cases = cases.dropna()
cases
```

[67]:

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

4	2020-01-24	Washington	1	0
5	2020-01-25	California	1	0
6	2020-01-25	Illinois	1	0
7	2020-01-25	Washington	1	0
8	2020-01-26	Arizona	1	0
9	2020-01-26	California	2	0
10	2020-01-26	Illinois	1	0
11	2020-01-26	Washington	1	0
12	2020-01-27	Arizona	1	0
13	2020-01-27	California	2	0
14	2020-01-27	Illinois	1	0
15	2020-01-27	Washington	1	0
16	2020-01-28	Arizona	1	0
17	2020-01-28	California	2	0
18	2020-01-28	Illinois	1	0
19	2020-01-28	Washington	1	0
20	2020-01-29	Arizona	1	0
21	2020-01-29	California	2	0
22	2020-01-29	Illinois	1	0
23	2020-01-29	Washington	1	0
24	2020-01-30	Arizona	1	0
25	2020-01-30	California	2	0
26	2020-01-30	Illinois	2	0
27	2020-01-30	Washington	1	0
28	2020-01-31	Arizona	1	0
29	2020-01-31	California	3	0
...	...	...	...	...
7464	2020-07-16	Mississippi	39797	1308
7465	2020-07-16	Missouri	32596	1145
7466	2020-07-16	Montana	2265	35
7467	2020-07-16	Nebraska	22178	307
7468	2020-07-16	Nevada	32024	627
7469	2020-07-16	New Hampshire	6139	395
7470	2020-07-16	New Jersey	178475	15665
7471	2020-07-16	New Mexico	16138	562
7472	2020-07-16	New York	409476	32133
7473	2020-07-16	North Carolina	93769	1617
7474	2020-07-16	North Dakota	4672	93
7475	2020-07-16	Northern Mariana Islands	37	2
7476	2020-07-16	Ohio	70601	3103
7477	2020-07-16	Oklahoma	23441	438
7478	2020-07-16	Oregon	13516	251
7479	2020-07-16	Pennsylvania	103169	7039
7480	2020-07-16	Puerto Rico	10574	172
7481	2020-07-16	Rhode Island	17711	988
7482	2020-07-16	South Carolina	64083	1070
7483	2020-07-16	South Dakota	7694	115

```
7484 2020-07-16                Tennessee   69827     786
7485 2020-07-16                    Texas  310489    3728
7486 2020-07-16                     Utah   31908     234
7487 2020-07-16                  Vermont    1325      56
7488 2020-07-16           Virgin Islands     263       6
7489 2020-07-16                 Virginia   74431    2007
7490 2020-07-16               Washington   46268    1492
7491 2020-07-16            West Virginia    4657      99
7492 2020-07-16                Wisconsin   43361     839
7493 2020-07-16                  Wyoming    2026      24

[7494 rows x 4 columns]
```

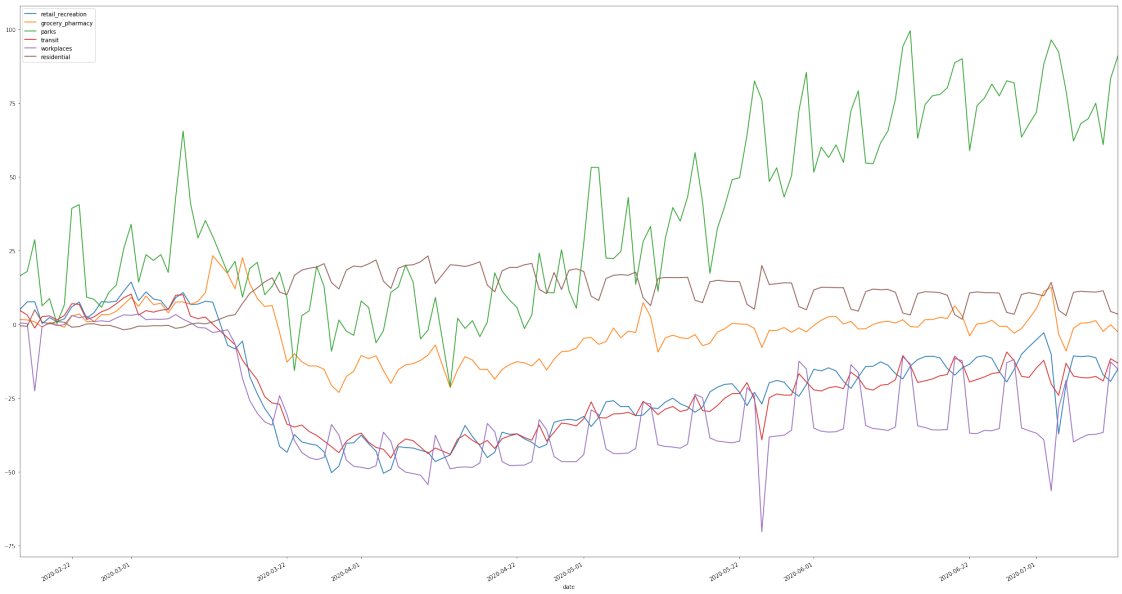
[68]:
```
mob = pd.read_csv("data/Global_Mobility_Report.csv",␣
 ↪names=['country_region_code', 'country', 'state',␣
 ↪'sub_region_2','iso_3166_2_code',
                                                    'census_fips_code',␣
 ↪'date', 'retail_recreation', 'grocery_pharmacy', 'parks',
                                                    'transit',␣
 ↪'workplaces', 'residential'], low_memory=False)
mobility = mob.loc[mob['country'] == 'United States']
mobility = mobility.drop(['country_region_code', 'census_fips_code',␣
 ↪'iso_3166_2_code', 'country', 'sub_region_2'], axis=1)
mobility = mobility.dropna()
mobility['date'] = mobility['date'].astype('datetime64[ns]')
mobility = mobility.reset_index().drop(['index'], axis=1)
mobility.describe()
```

[68]:
```
              state                 date retail_recreation grocery_pharmacy  \
count         75701                75701             75701            75701
unique           51                  149               146              139
top     California  2020-02-21 00:00:00                 4                1
freq           5577                  604              1360             2735
first           NaN  2020-02-15 00:00:00               NaN              NaN
last            NaN  2020-07-12 00:00:00               NaN              NaN

        parks transit workplaces residential
count   75701   75701      75701       75701
unique    514     192        111          45
top         6       2          2           0
freq      805    1405       3136        5125
first     NaN     NaN        NaN         NaN
last      NaN     NaN        NaN         NaN
```

[69]:
```
# Visualize the mobility data over time
mob_date = mobility.sort_index(level=['date'])
mob_date['retail_recreation'] = mob_date['retail_recreation'].astype('double')
```

```
mob_date['grocery_pharmacy'] = mob_date['grocery_pharmacy'].astype('double')
mob_date['parks'] = mob_date['parks'].astype('double')
mob_date['transit'] = mob_date['transit'].astype('double')
mob_date['workplaces'] = mob_date['workplaces'].astype('double')
mob_date['residential'] = mob_date['residential'].astype('double')
mob_date = mob_date.groupby(['date']).mean()
# Remove any outliers
z_scores = zscore(mob_date)
abs_z_scores = np.abs(z_scores)
filtered_entries = (abs_z_scores < 3).all(axis=1)
mob_date = mob_date[filtered_entries]
# Plot the overall change in mobility over time
mob_date.plot(figsize=(35,20))
```

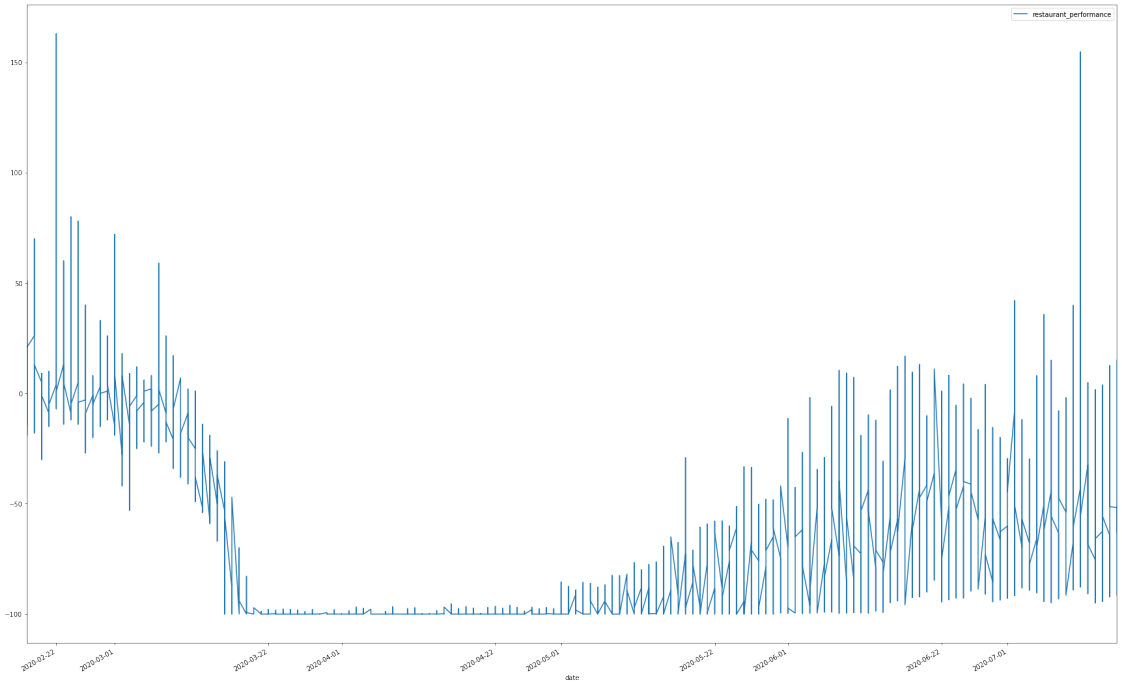
[69]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99dad7d30>



[70]:
```
# Restaurant data
rest = pd.read_csv("data/YoY_Seated_Diner_Data.csv")
restaurants = rest[8:57]
restaurants = restaurants.drop([11, 12, 28, 36, 45, 46, 47, 39, 42, 9, 22, 53])
restaurants = restaurants.reset_index().drop(['index', 'Type'], axis=1)
restaurants = restaurants.melt(id_vars=["Name"], var_name="date",␣
 ↪value_name="restaurant_performance")
restaurants = restaurants.sort_values(['Name', 'date'], ascending=[True, True])
def add_year(date_in_some_format):
    date_as_string = str(date_in_some_format)
    new_date = date_as_string + '/2020'
```

```
    return new_date
restaurants['date'] = restaurants['date'].apply(add_year)
restaurants['date'] = restaurants['date'].astype('datetime64[ns]')
restaurants = restaurants.rename({'Name':'state'}, axis='columns')
restaurants.plot(x='date', y='restaurant_performance', figsize=(30,20))
```

[70]: <matplotlib.axes._subplots.AxesSubplot at 0x1a99e8c8da0>



[71]:
```
restaurants.describe()
```

[71]:
```
       restaurant_performance
count             5550.000000
mean                -68.545553
std                  36.498080
min                -100.000000
25%                -100.000000
50%                 -81.080000
75%                 -49.007500
max                 163.000000
```

[72]:
```
# Merge the movbility, case and restaurant data
all_data = pd.merge(mobility, cases, on=["state", "date"], how="left")
all_data = all_data.dropna()
all_data = pd.merge(all_data, restaurants, on=["state", "date"], how="left")
all_data = all_data.dropna()
```

```
all_data = all_data.reset_index().drop(['index'], axis=1)
all_data['date'] = all_data['date'].astype('datetime64[ns]')

all_data['retail_recreation'] = all_data['retail_recreation'].astype('int')
all_data['grocery_pharmacy'] = all_data['grocery_pharmacy'].astype('int')
all_data['parks'] = all_data['parks'].astype('int')
all_data['transit'] = all_data['transit'].astype('int')
all_data['workplaces'] = all_data['workplaces'].astype('int')
all_data['residential'] = all_data['residential'].astype('int')

all_data_new = all_data.copy()

# Print
all_data
```

[72]:

	state	date	retail_recreation	grocery_pharmacy	parks	\
0	Alabama	2020-03-13	7	32	26	
1	Alabama	2020-03-14	1	28	55	
2	Alabama	2020-03-15	-7	16	16	
3	Alabama	2020-03-16	-2	24	22	
4	Alabama	2020-03-17	-11	17	25	
5	Alabama	2020-03-18	-13	13	39	
6	Alabama	2020-03-19	-20	15	27	
7	Alabama	2020-03-20	-27	10	7	
8	Alabama	2020-03-21	-35	5	10	
9	Alabama	2020-03-22	-40	-7	2	
10	Alabama	2020-03-23	-33	-6	-11	
11	Alabama	2020-03-24	-32	-5	4	
12	Alabama	2020-03-25	-28	-4	36	
13	Alabama	2020-03-26	-30	-3	32	
14	Alabama	2020-03-27	-30	-2	30	
15	Alabama	2020-03-28	-36	-2	22	
16	Alabama	2020-03-29	-41	-13	19	
17	Alabama	2020-03-30	-33	-10	10	
18	Alabama	2020-03-31	-36	-11	-23	
19	Alabama	2020-04-01	-25	5	19	
20	Alabama	2020-04-02	-32	3	12	
21	Alabama	2020-04-03	-29	12	17	
22	Alabama	2020-04-04	-41	8	4	
23	Alabama	2020-04-05	-50	-18	-1	
24	Alabama	2020-04-06	-37	-10	3	
25	Alabama	2020-04-07	-39	-11	-15	
26	Alabama	2020-04-08	-36	-9	-1	
27	Alabama	2020-04-09	-35	-4	5	
28	Alabama	2020-04-10	-34	-4	4	
29	Alabama	2020-04-11	-39	3	11	
...	...	...	...	...	...	

```
61131  Wisconsin 2020-04-28               -41              -16     26
61132  Wisconsin 2020-04-29               -41              -17    -19
61133  Wisconsin 2020-04-30               -33              -11    114
61134  Wisconsin 2020-05-01               -35              -15    159
61135  Wisconsin 2020-05-04               -31              -14     81
61136  Wisconsin 2020-05-05               -32              -12     70
61137  Wisconsin 2020-05-07               -32               -8    144
61138  Wisconsin 2020-05-08               -37              -11     49
61139  Wisconsin 2020-05-09               -30                6    109
61140  Wisconsin 2020-05-11               -34              -13     87
61141  Wisconsin 2020-05-13               -28              -10    104
61142  Wisconsin 2020-05-14               -32              -14     89
61143  Wisconsin 2020-05-15               -30              -10    205
61144  Wisconsin 2020-05-17               -39              -27    -37
61145  Wisconsin 2020-05-18               -31              -15     -8
61146  Wisconsin 2020-05-19               -24              -10     83
61147  Wisconsin 2020-05-22               -22               -7    177
61148  Wisconsin 2020-05-28               -21              -13     30
61149  Wisconsin 2020-05-29               -17               -9    158
61150  Wisconsin 2020-06-01               -10               -5    107
61151  Wisconsin 2020-06-05                -9               -6    173
61152  Wisconsin 2020-06-09               -13               -8    122
61153  Wisconsin 2020-06-19                -8               -3    188
61154  Wisconsin 2020-06-22               -11               -7     79
61155  Wisconsin 2020-06-26               -13               -9    111
61156  Wisconsin 2020-06-29                -7               -5    112
61157  Wisconsin 2020-07-03                -5                5    194
61158  Wisconsin 2020-07-07                -7               -6    123
61159  Wisconsin 2020-07-09                -6               -3    103
61160  Wisconsin 2020-07-10                -6               -5    186

       transit  workplaces  residential    cases  deaths  \
0            7          -2            0      6.0     0.0
1           12           4            0     12.0     0.0
2            6          -4            2     23.0     0.0
3            2         -10            4     29.0     0.0
4           -1         -17            7     39.0     0.0
5           -3         -22            8     51.0     0.0
6           -8         -25           10     78.0     0.0
7          -14         -29           14    106.0     0.0
8          -13         -16            8    131.0     0.0
9          -22         -27            9    157.0     0.0
10         -22         -33           14    196.0     0.0
11         -18         -34           15    242.0     0.0
12         -15         -36           13    386.0     1.0
13         -19         -36           14    538.0     3.0
14         -24         -37           16    639.0     4.0
```

15	-25	-22	9	720.0	4.0
16	-30	-32	9	830.0	5.0
17	-25	-38	14	947.0	11.0
18	-26	-40	17	999.0	14.0
19	-20	-39	14	1108.0	28.0
20	-25	-39	16	1270.0	32.0
21	-29	-40	16	1535.0	38.0
22	-33	-26	11	1632.0	44.0
23	-40	-36	12	1840.0	45.0
24	-29	-41	16	2005.0	53.0
25	-31	-42	18	2197.0	64.0
26	-30	-43	17	2498.0	67.0
27	-31	-43	18	2838.0	78.0
28	-37	-48	20	3008.0	80.0
29	-35	-29	10	3262.0	93.0
...	...	...	...	...	...
61131	-33	-39	13	6289.0	300.0
61132	-35	-40	19	6520.0	308.0
61133	-21	-39	16	6854.0	316.0
61134	-26	-40	16	7314.0	327.0
61135	-23	-37	15	8236.0	340.0
61136	-24	-37	16	8566.0	353.0
61137	-24	-37	16	9215.0	374.0
61138	-19	-38	17	9590.0	384.0
61139	-14	-21	7	9939.0	398.0
61140	-27	-36	15	10418.0	409.0
61141	-17	-36	15	10903.0	421.0
61142	-20	-36	15	11280.0	434.0
61143	-14	-37	13	11854.0	445.0
61144	-37	-21	9	12571.0	453.0
61145	-36	-34	15	12722.0	459.0
61146	-26	-33	14	13001.0	467.0
61147	-12	-37	11	14557.0	496.0
61148	-19	-31	14	17211.0	550.0
61149	16	-30	11	17784.0	569.0
61150	18	-29	10	18556.0	597.0
61151	47	-30	9	20427.0	634.0
61152	28	-29	11	21435.0	662.0
61153	27	-31	7	26913.0	736.0
61154	20	-31	10	27849.0	751.0
61155	34	-31	10	29718.0	775.0
61156	8	-31	9	31103.0	786.0
61157	25	-57	11	33565.0	804.0
61158	24	-33	10	35834.0	817.0
61159	38	-31	10	37358.0	821.0
61160	62	-32	8	38414.0	825.0

	restaurant_performance
0	-24.00
1	-28.00
2	-38.00
3	-57.00
4	-74.00
5	-78.00
6	-96.77
7	-100.00
8	-100.00
9	-100.00
10	-100.00
11	-100.00
12	-100.00
13	-100.00
14	-100.00
15	-100.00
16	-100.00
17	-100.00
18	-100.00
19	-100.00
20	-100.00
21	-100.00
22	-100.00
23	-100.00
24	-100.00
25	-100.00
26	-100.00
27	-100.00
28	-100.00
29	-100.00
...	...
61131	-100.00
61132	-100.00
61133	-100.00
61134	-100.00
61135	-100.00
61136	-100.00
61137	-100.00
61138	-100.00
61139	-100.00
61140	-100.00
61141	-100.00
61142	-100.00
61143	-99.37
61144	-98.49
61145	-98.71

```
61146                  -97.43
61147                  -88.70
61148                  -83.38
61149                  -80.28
61150                  -85.70
61151                  -72.00
61152                  -78.05
61153                  -50.58
61154                  -60.24
61155                  -57.26
61156                  -67.14
61157                  -49.03
61158                  -58.37
61159                  -63.04
61160                  -45.45

[61161 rows x 11 columns]
```

[73]:
```
all_normalized = all_data.copy()

all_normalized['cases_diff'] = (all_normalized.groupby('date')['cases'].apply(pd.
 ↪Series.pct_change))

# Normalize the values

scaler = preprocessing.StandardScaler()
all_normalized[['cases', 'deaths', 'restaurant_performance',␣
 ↪'retail_recreation', 'grocery_pharmacy', 'parks', 'transit', 'workplaces',␣
 ↪'residential', 'cases', 'deaths', 'restaurant_performance']] = scaler.
 ↪fit_transform(all_normalized[['cases', 'deaths', 'restaurant_performance',␣
 ↪'retail_recreation', 'grocery_pharmacy', 'parks', 'transit', 'workplaces',␣
 ↪'residential', 'cases', 'deaths', 'restaurant_performance']])

#drop Nan values
all_normalized = all_normalized.replace([np.inf, -np.inf], np.nan)
all_normalized = all_normalized.dropna()

all_normalized = all_normalized.groupby(['date']).mean()
#all_normalized = all_normalized.loc[all_normalized['cases'] > -0.5]
all_normalized
```

[73]:
```
            retail_recreation  grocery_pharmacy     parks   transit  \
date
2020-02-18           1.283461          0.280434 -0.280377  1.172683
2020-02-19           1.299567          0.288020 -0.337517  1.129320
2020-02-20           1.334336          0.320521 -0.353005  1.129598
2020-02-21           1.348272          0.273682 -0.306777  1.184425
```

2020-02-22	1.486680	0.453104	-0.065465	1.201621
2020-02-23	1.640029	0.515321	-0.049337	1.261562
2020-02-24	1.370758	0.429771	-0.294880	1.104315
2020-02-25	1.433415	0.378990	-0.248552	1.150720
2020-02-26	1.620922	0.552107	-0.290068	1.214174
2020-02-27	1.659027	0.638469	-0.151852	1.285118
2020-02-28	1.625691	0.675234	-0.236979	1.329643
2020-02-29	1.800597	0.839432	-0.075467	1.365449
2020-03-01	1.907730	0.878092	-0.105445	1.385517
2020-03-02	1.638059	0.799053	-0.324537	1.136270
2020-03-03	1.745237	1.014306	-0.151289	1.195233
2020-03-04	1.621185	0.778934	-0.204587	1.164788
2020-03-05	1.606607	0.822732	-0.162393	1.190186
2020-03-06	1.444588	0.593783	-0.261244	1.204675
2020-03-07	1.629375	0.817736	0.122274	1.385279
2020-03-08	1.717161	0.787410	0.451220	1.381989
2020-03-09	1.553688	0.770459	0.163173	1.113102
2020-03-10	1.549740	0.827743	-0.057224	1.081415
2020-03-11	1.608842	1.059391	0.051620	1.110373
2020-03-12	1.590219	1.973148	-0.044509	0.997346
2020-03-13	1.309392	2.286209	-0.258484	0.859018
2020-03-14	0.863698	1.548603	-0.212065	0.831929
2020-03-15	0.784131	1.129257	-0.165883	0.724838
2020-03-16	0.919410	1.909871	-0.370584	0.497640
2020-03-17	0.304342	1.253373	-0.219051	0.354784
2020-03-18	0.025161	0.911077	-0.199618	0.223730
...	...	...	...	...
2020-06-13	0.239129	0.343727	0.905099	0.532832
2020-06-14	0.441857	0.160894	0.988091	0.411731
2020-06-15	0.560374	0.154672	0.407475	0.165099
2020-06-16	0.613071	0.346746	0.589534	0.190639
2020-06-17	0.620947	0.352177	0.657489	0.219049
2020-06-18	0.594515	0.409039	0.632479	0.248488
2020-06-19	0.417206	0.378895	0.663401	0.267343
2020-06-20	0.306421	0.697851	0.801364	0.514577
2020-06-21	0.405987	0.419668	0.789781	0.422289
2020-06-22	0.487601	-0.055185	0.351431	0.160846
2020-06-23	0.616483	0.248137	0.577388	0.203819
2020-06-24	0.644225	0.268863	0.643400	0.245740
2020-06-25	0.587660	0.331605	0.687217	0.267250
2020-06-26	0.369481	0.184215	0.616633	0.290019
2020-06-27	0.185488	0.177087	0.685777	0.565706
2020-06-28	0.378295	-0.017798	0.691002	0.452166
2020-06-29	0.649688	0.124621	0.448964	0.236509
2020-06-30	0.782897	0.361739	0.494535	0.221353
2020-07-01	0.896806	0.605101	0.535135	0.348985
2020-07-02	1.026673	1.039432	0.779763	0.451855

2020-07-03	0.660530	1.139697	0.892785	0.132076	
2020-07-04	-0.684974	-0.019903	0.847514	-0.015330	
2020-07-05	0.136493	-0.468238	0.614000	0.407156	
2020-07-06	0.625851	0.119528	0.373255	0.237256	
2020-07-07	0.612282	0.250406	0.456594	0.208335	
2020-07-08	0.638444	0.278349	0.527146	0.225652	
2020-07-09	0.613875	0.338515	0.592858	0.244169	
2020-07-10	0.310128	0.051275	0.340182	0.171184	
2020-07-11	0.191998	0.205760	0.715225	0.461912	
2020-07-12	0.404532	0.022893	0.834255	0.415298	

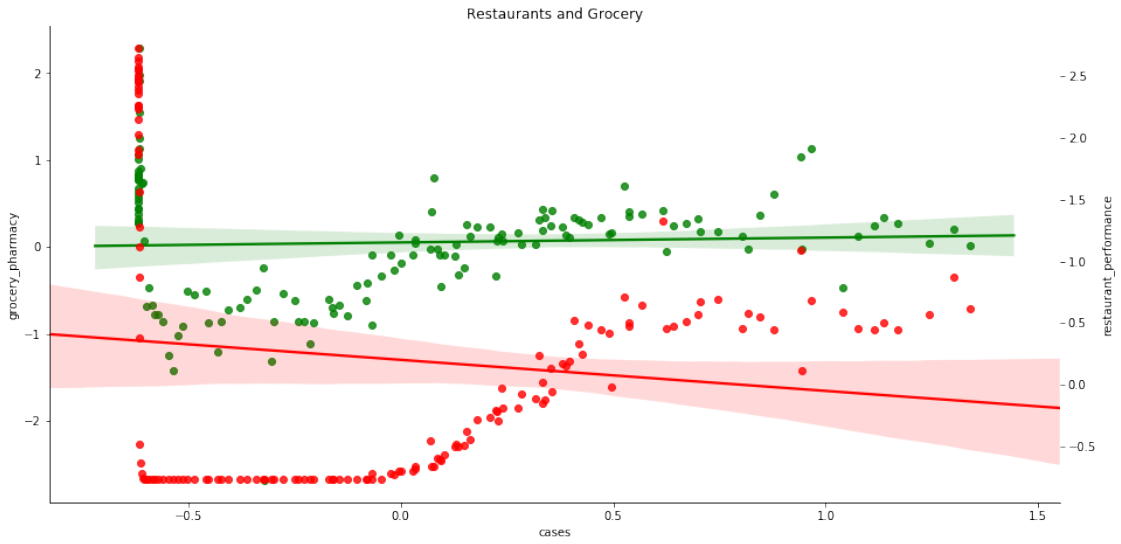
date	workplaces	residential	cases	deaths	\
2020-02-18	1.808925	-1.539870	-0.617300	-0.458610	
2020-02-19	1.856543	-1.550126	-0.617299	-0.458610	
2020-02-20	1.827480	-1.532035	-0.617295	-0.458610	
2020-02-21	1.871607	-1.590621	-0.617285	-0.458610	
2020-02-22	1.980912	-1.687321	-0.617284	-0.458610	
2020-02-23	1.968618	-1.675852	-0.617284	-0.458610	
2020-02-24	2.087964	-1.645881	-0.617250	-0.458610	
2020-02-25	1.986538	-1.634892	-0.617256	-0.458610	
2020-02-26	1.964996	-1.684627	-0.617204	-0.458610	
2020-02-27	1.970542	-1.718290	-0.617202	-0.458610	
2020-02-28	1.992181	-1.761345	-0.617204	-0.458610	
2020-02-29	2.025512	-1.844627	-0.617197	-0.458588	
2020-03-01	2.006022	-1.747040	-0.617209	-0.458563	
2020-03-02	2.052228	-1.680498	-0.617202	-0.458511	
2020-03-03	1.954764	-1.667613	-0.617182	-0.458452	
2020-03-04	1.951574	-1.626667	-0.617147	-0.458433	
2020-03-05	1.943388	-1.644894	-0.617109	-0.458450	
2020-03-06	1.942444	-1.617436	-0.617085	-0.458436	
2020-03-07	2.030693	-1.744933	-0.617012	-0.458410	
2020-03-08	1.939734	-1.690981	-0.616948	-0.458402	
2020-03-09	1.878547	-1.565158	-0.616839	-0.458347	
2020-03-10	1.789516	-1.509804	-0.616694	-0.458312	
2020-03-11	1.777465	-1.556080	-0.616576	-0.458268	
2020-03-12	1.684265	-1.461035	-0.616339	-0.458216	
2020-03-13	1.509470	-1.216094	-0.616040	-0.458179	
2020-03-14	1.752187	-1.209403	-0.615725	-0.458098	
2020-03-15	1.471135	-1.140129	-0.615379	-0.458058	
2020-03-16	0.798085	-0.631391	-0.614854	-0.457805	
2020-03-17	0.363861	-0.167900	-0.613947	-0.457546	
2020-03-18	0.126051	0.073740	-0.612637	-0.457163	
...	...	...	...	...	
2020-06-13	1.220929	-1.044817	0.406685	0.318710	
2020-06-14	1.081677	-1.134653	0.495764	0.414423	
2020-06-15	-0.111239	-0.168183	0.488982	0.402653	

2020-06-16	-0.148286	-0.091422	0.471489	0.375261
2020-06-17	-0.192102	-0.105762	0.537381	0.432615
2020-06-18	-0.194765	-0.129847	0.537149	0.412857
2020-06-19	-0.183082	-0.243752	0.566611	0.436169
2020-06-20	1.191559	-1.116955	0.524925	0.350887
2020-06-21	1.144232	-1.326372	0.616551	0.449157
2020-06-22	-0.252917	-0.152687	0.624589	0.447944
2020-06-23	-0.260588	-0.100154	0.641843	0.439353
2020-06-24	-0.195791	-0.139121	0.671696	0.453747
2020-06-25	-0.211722	-0.127852	0.699502	0.467175
2020-06-26	-0.164489	-0.146950	0.703686	0.448337
2020-06-27	1.106370	-1.018262	0.745905	0.444617
2020-06-28	1.166815	-1.098015	0.816340	0.504789
2020-06-29	-0.152470	-0.221423	0.802270	0.464353
2020-06-30	-0.206384	-0.138888	0.843709	0.492731
2020-07-01	-0.251810	-0.199905	0.878187	0.499155
2020-07-02	-0.371721	-0.274630	0.940405	0.498065
2020-07-03	-1.338292	0.325338	0.965652	0.491149
2020-07-04	0.240778	-0.908826	0.942921	0.387366
2020-07-05	0.768291	-1.153152	1.039893	0.517934
2020-07-06	-0.418996	-0.118223	1.074987	0.531769
2020-07-07	-0.341905	-0.066077	1.113464	0.504849
2020-07-08	-0.272685	-0.105883	1.169992	0.564270
2020-07-09	-0.271910	-0.113601	1.136271	0.473416
2020-07-10	-0.239500	-0.041782	1.242127	0.566975
2020-07-11	1.122047	-0.980577	1.299856	0.534479
2020-07-12	0.982589	-1.099578	1.340897	0.566258

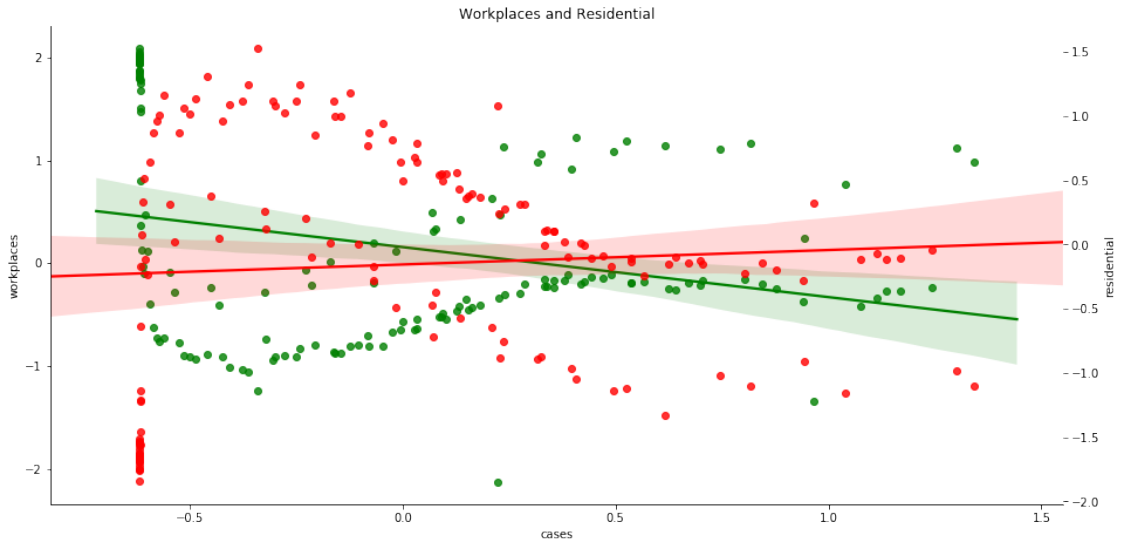
date	restaurant_performance	cases_diff
2020-02-18	2.493814	0.086108
2020-02-19	2.556566	0.085519
2020-02-20	2.544220	0.098848
2020-02-21	2.450074	0.101562
2020-02-22	2.647127	0.109542
2020-02-23	2.722611	0.115307
2020-02-24	2.471395	0.128276
2020-02-25	2.570777	0.149678
2020-02-26	2.462273	0.246053
2020-02-27	2.459749	0.246053
2020-02-28	2.504247	0.286694
2020-02-29	2.617823	0.325505
2020-03-01	2.411483	0.296253
2020-03-02	2.260353	0.268128
2020-03-03	2.148240	0.229766
2020-03-04	2.266334	0.314013
2020-03-05	2.238873	0.403166

```
2020-03-06                2.256888    0.232339
2020-03-07                2.354807    0.485203
2020-03-08                2.390204    0.402355
2020-03-09                2.022875    0.276265
2020-03-10                1.907651    0.262816
2020-03-11                1.863757    0.345708
2020-03-12                1.562801    0.316610
2020-03-13                1.280161    0.268005
2020-03-14                1.115176    0.245642
2020-03-15                0.869551    0.255595
2020-03-16                0.381902    0.229950
2020-03-17               -0.482848    0.271094
2020-03-18               -0.629577    0.329480
...                            ...         ...
2020-06-13                0.521345    0.710108
2020-06-14               -0.012705    0.680980
2020-06-15                0.417295    0.632149
2020-06-16                0.444849    0.630650
2020-06-17                0.479530    0.632933
2020-06-18                0.502608    0.617754
2020-06-19                0.641136    0.595383
2020-06-20                0.707971    0.632039
2020-06-21                1.330215    0.608315
2020-06-22                0.456473    0.576644
2020-06-23                0.472889    0.592380
2020-06-24                0.509929    0.573485
2020-06-25                0.572696    0.582975
2020-06-26                0.677424    0.556593
2020-06-27                0.691987    0.600526
2020-06-28                0.580977    0.579203
2020-06-29                0.460628    0.548781
2020-06-30                0.554136    0.549600
2020-07-01                0.448352    0.545578
2020-07-02                1.087080    0.565185
2020-07-03                0.678577    0.560560
2020-07-04                0.117753    0.580040
2020-07-05                0.585819    0.526661
2020-07-06                0.457554    0.516373
2020-07-07                0.442591    0.507467
2020-07-08                0.450493    0.493739
2020-07-09                0.502358    0.484024
2020-07-10                0.574024    0.464597
2020-07-11                0.874811    0.506999
2020-07-12                0.620204    0.500872

[146 rows x 10 columns]
```

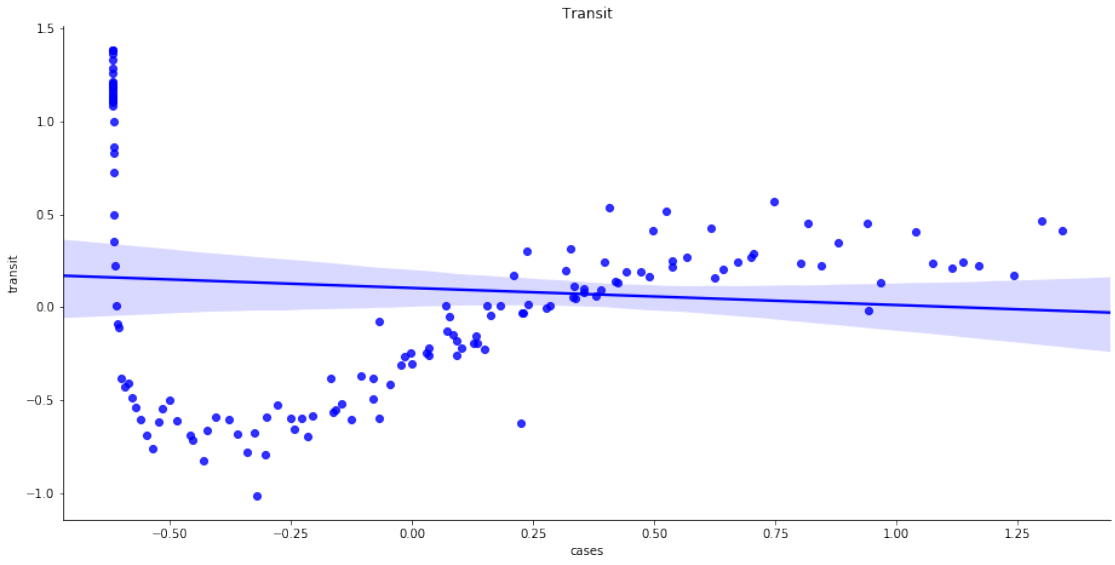
[74]:
```
# Plot the relationship between number of cases and restaurant performance and␣
 ↪grocery/pharmacy mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(all_normalized['cases'], all_normalized['grocery_pharmacy'], ax=ax,␣
 ↪color='g')
ax2 = ax.twinx()
sns.regplot(all_normalized['cases'], all_normalized['restaurant_performance'],␣
 ↪ax=ax2, color='r').set_title('Restaurants and Grocery')
sns.despine()
```



[75]:
```
# Plot the relationship between number of cases and workplaces and residential␣
 ↪mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(all_normalized['cases'], all_normalized['workplaces'], ax=ax,␣
 ↪color='g')
ax2 = ax.twinx()
sns.regplot(all_normalized['cases'], all_normalized['residential'], ax=ax2,␣
 ↪color='r').set_title('Workplaces and Residential')
sns.despine()
```

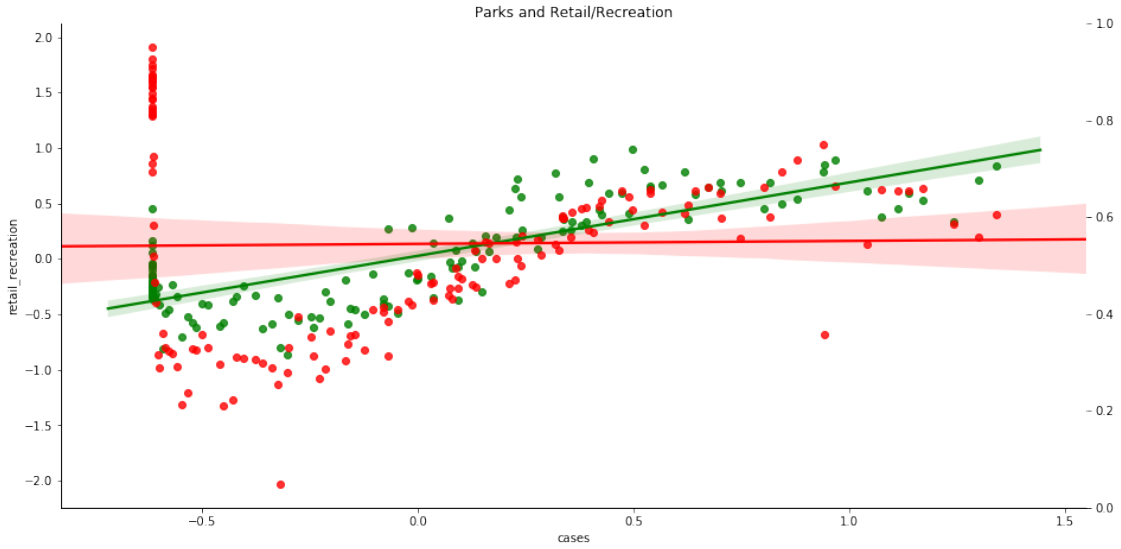


[76]:
```
# Plot the relationship between number of cases and transit mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(all_normalized['cases'], all_normalized['transit'], ax=ax,␣
↪color='b').set_title('Transit')
sns.despine()
```



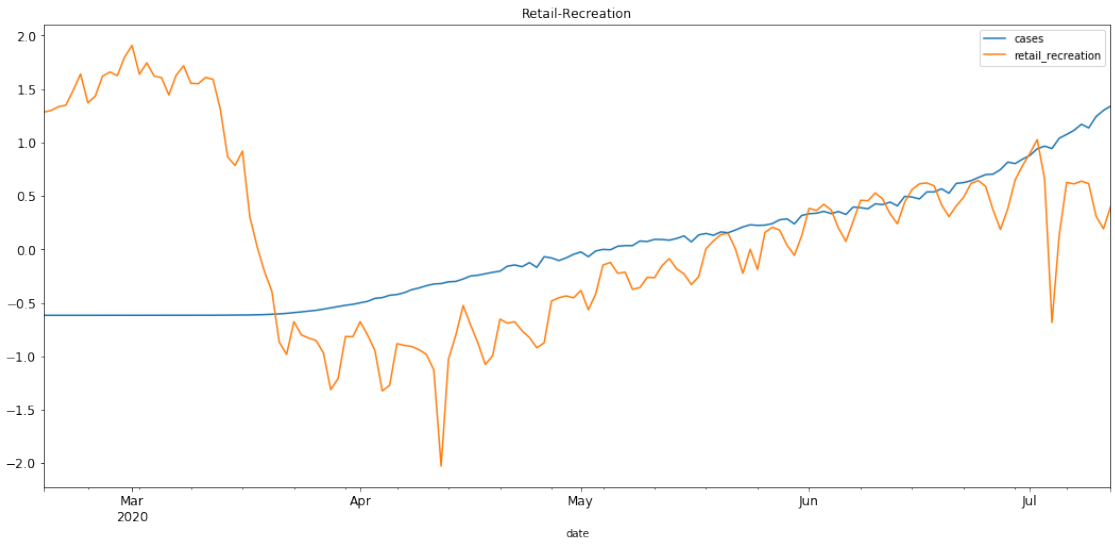
[77]:
```
# Plot the relationship between number of cases and parks and retail/recreation␣
↪mobility
fig, ax = plt.subplots()
```

```
fig.set_size_inches(15.5, 7.5)
sns.regplot(all_normalized['cases'], all_normalized['parks'], ax=ax, color='g')
ax2 = ax.twinx()
sns.regplot(all_normalized['cases'], all_normalized['retail_recreation'], ax=ax,␣
↪color='r').set_title('Parks and Retail/Recreation')
sns.despine()
```



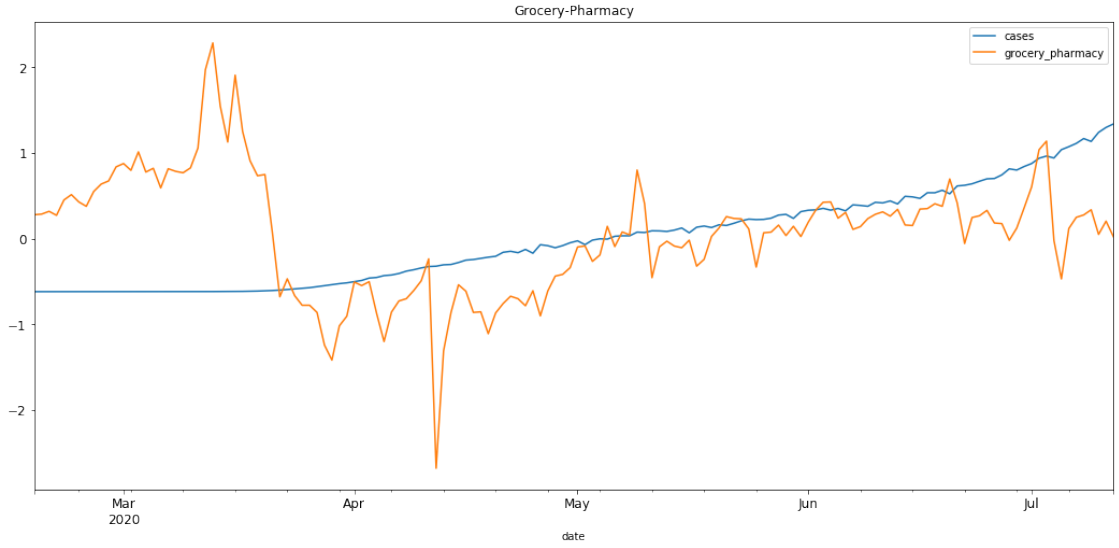
[78]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'retail_recreation'],␣
↪figsize=(18,8), fontsize=12, title="Retail-Recreation")
```

[78]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a2fb48d0>



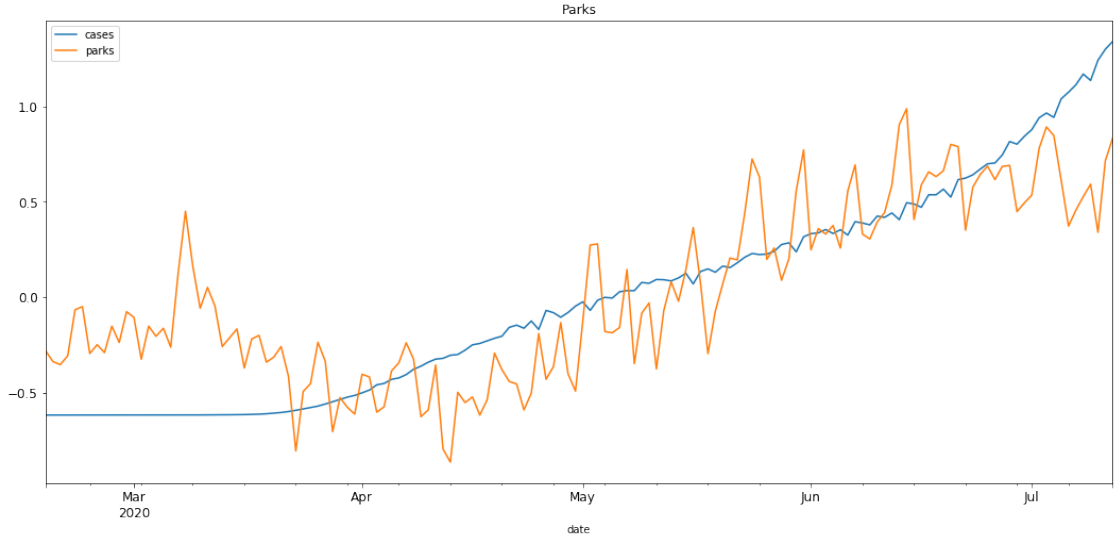
[79]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'grocery_pharmacy'],␣
↪figsize=(18,8), fontsize=12, title="Grocery-Pharmacy")
```

[79]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a3073a20>



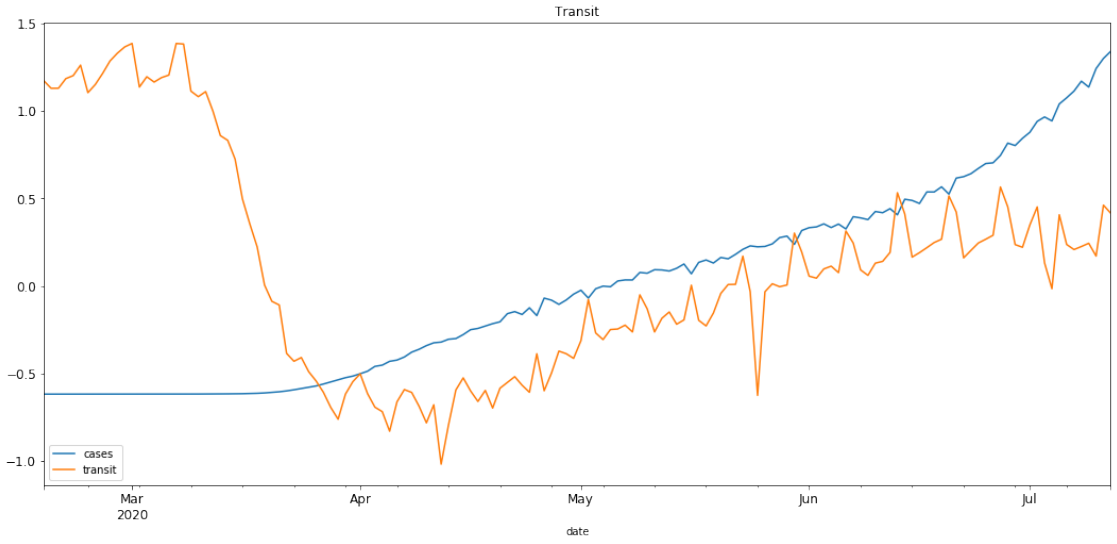
[80]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'parks'],␣
↪figsize=(18,8), fontsize=12, title="Parks")
```

[80]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a310a9e8>



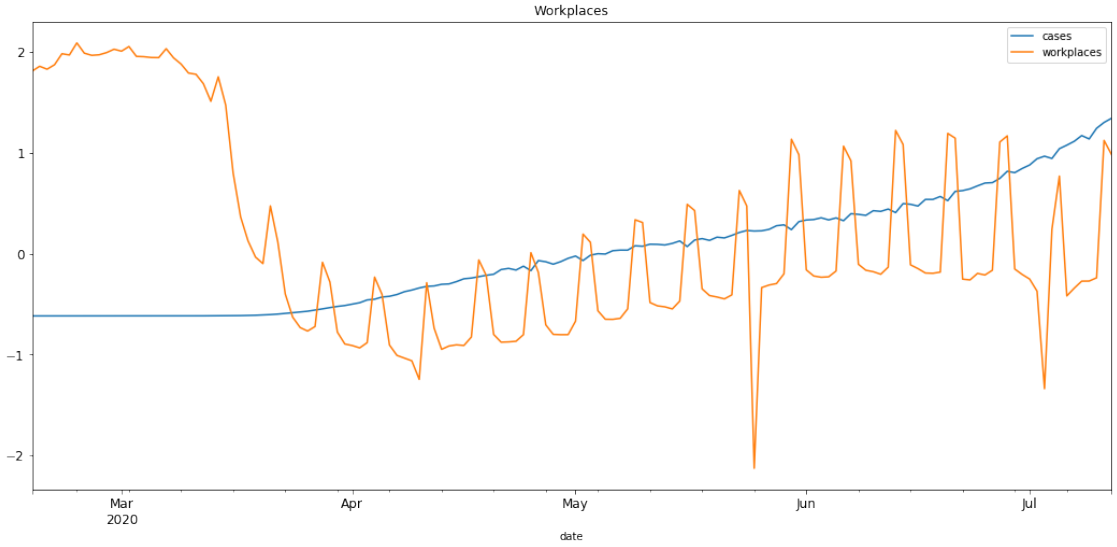
[81]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'transit'],␣
↪figsize=(18,8), fontsize=12, title="Transit")
```

[81]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a49cbbe0>



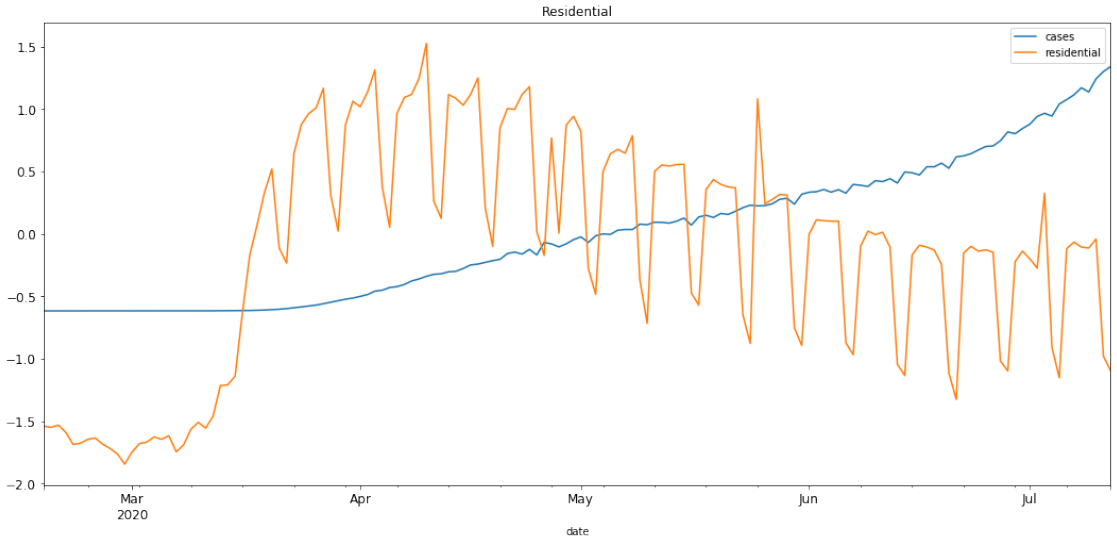
[82]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'workplaces'],␣
↪figsize=(18,8), fontsize=12, title="Workplaces")
```

[82]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a49cb358>



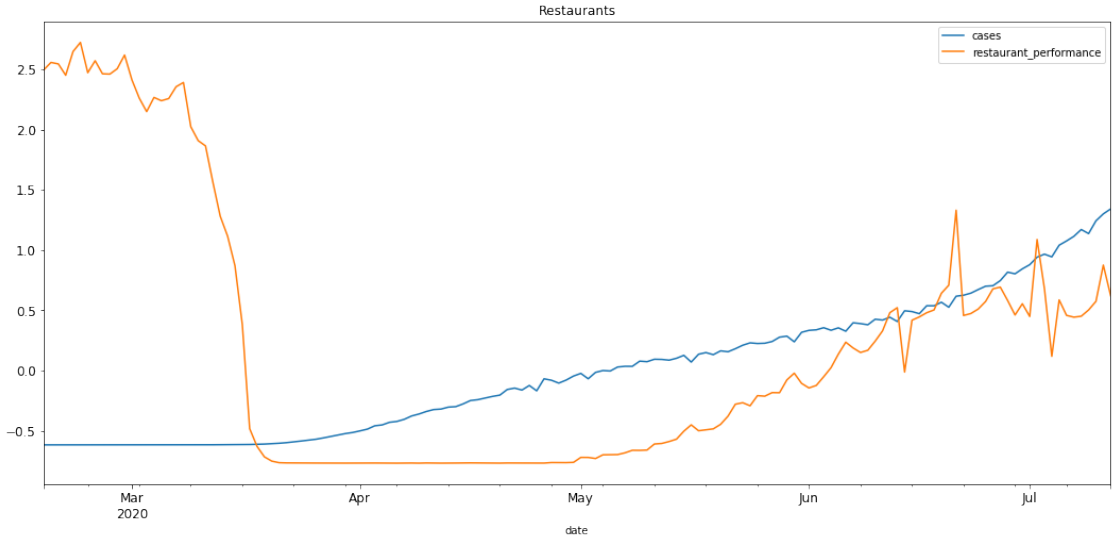
[83]:
```
all_normalized.reset_index().plot(x='date', y=['cases', 'residential'],␣
↪figsize=(18,8), fontsize=12, title="Residential")
```

[83]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a50fbe80>



[84]:
```
all_normalized.reset_index().plot(x='date', y=['cases',␣
↪'restaurant_performance'], figsize=(18,8), fontsize=12, title="Restaurants")
```

[84]: <matplotlib.axes._subplots.AxesSubplot at 0x1a9a5ab9860>



[85]:
```
#All Data by state
# Create new dataframe from the percent change in mobility in each area and␣
 ↪group by state
change = all_data_new.sort_index(level=['state', 'date'])

change['diff_cases'] = (change.groupby('state')['cases'].apply(pd.Series.diff))

#drop Nan values
change = change.replace([np.inf, -np.inf], np.nan)
change = change.dropna()

change['retail_recreation'] = change['retail_recreation'].astype(int)
change['grocery_pharmacy'] = change['grocery_pharmacy'].astype(int)
change['parks'] = change['parks'].astype(int)
change['transit'] = change['transit'].astype(int)
change['workplaces'] = change['workplaces'].astype(int)
change['residential'] = change['residential'].astype(int)

state = change.groupby(['state']).mean()

state = state.sort_values(['diff_cases'])
state
```

[85]:

state	retail_recreation	grocery_pharmacy	parks	\
Hawaii	-43.011609	-23.976783	-48.620232	
Oregon	-19.145726	-0.141136	50.118761	
Ohio	-23.394988	-1.938544	80.799523	
Maryland	-28.569112	-11.598456	54.080309	
New Mexico	-21.941176	3.764706	11.664799	
Kentucky	-22.298507	2.334992	52.414594	
Washington	-18.758506	-0.407469	54.171784	
California	-30.258778	-4.725494	5.504755	
Georgia	-18.457043	-1.129089	17.661579	
Colorado	-23.745086	-4.069410	44.149877	
Missouri	-20.720539	0.623737	49.238215	
Wisconsin	-23.731921	-3.780131	65.020453	
Indiana	-21.737895	-1.756842	52.177895	
Oklahoma	-12.183612	4.198786	28.531108	
Utah	-14.602312	5.936416	79.810405	
Kansas	-18.412844	0.434862	71.816514	
South Carolina	-15.807374	1.196440	20.742530	
Rhode Island	-24.178654	-8.498840	71.387471	
Virginia	-27.847003	-6.725552	31.194006	
Pennsylvania	-28.531631	-7.769055	54.447790	
Tennessee	-18.175775	-0.161743	28.394387	
North Carolina	-22.133333	-3.893424	26.142857	

Minnesota	-26.999024	-3.047805	82.602927
Nebraska	-16.379175	1.049116	73.660118
Alabama	-18.501385	2.178209	20.024931
Michigan	-31.681339	-9.050837	88.875387
Texas	-17.329794	-3.852535	7.623322
Louisiana	-24.215517	-2.056034	-4.045259
Florida	-25.020824	-9.041863	-13.634392
Connecticut	-24.851648	-6.894231	49.265110
Nevada	-25.818408	-4.106965	-5.833333
Massachusetts	-25.233792	-8.747217	58.874263
Arizona	-17.071385	-2.257713	2.011494
Illinois	-23.171653	0.786660	43.475723
New Jersey	-38.354646	-7.741490	51.931923
District of Columbia	-53.826772	-22.645669	-40.566929
New York	-37.175400	-7.097889	51.308588

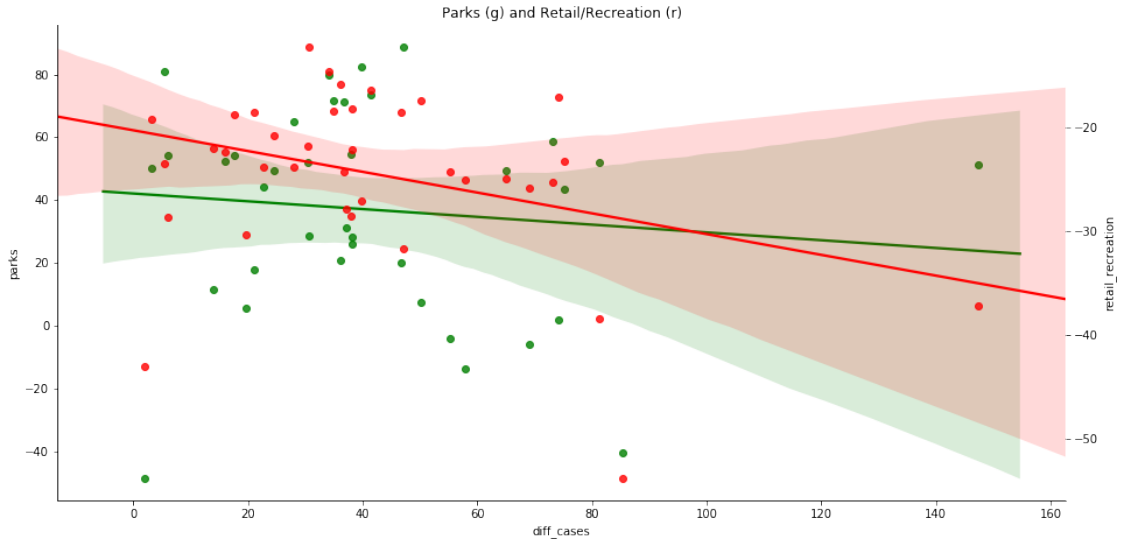
state	transit	workplaces	residential	cases	\
Hawaii	-62.908789	-35.747927	14.431177	548.081260	
Oregon	-20.030407	-30.682157	9.322433	3344.050488	
Ohio	-15.900955	-34.437947	12.047733	21651.952864	
Maryland	-35.928185	-39.963707	16.210039	28731.347490	
New Mexico	-23.900093	-33.004669	11.751634	5527.202614	
Kentucky	-20.936982	-34.126036	11.709784	6160.822554	
Washington	-17.663900	-30.486307	9.756017	14243.557261	
California	-28.161668	-31.020483	11.820227	78726.008778	
Georgia	-23.830789	-31.781073	11.164850	31831.810292	
Colorado	-24.300369	-35.988943	12.468673	16584.372850	
Missouri	-15.279461	-32.491582	10.946128	9385.941077	
Wisconsin	-21.537619	-27.088386	10.994156	8714.373265	
Indiana	-14.106316	-32.435789	11.845263	17793.322105	
Oklahoma	-4.558422	-27.280728	8.679818	5168.051593	
Utah	-16.360694	-30.015029	9.341040	7204.515607	
Kansas	-4.691743	-30.594495	10.071560	5807.486239	
South Carolina	-12.137317	-29.339479	9.947870	11955.863318	
Rhode Island	-43.095128	-35.459397	13.856148	7512.436195	
Virginia	-35.184017	-38.632492	14.749737	26260.774448	
Pennsylvania	-29.771723	-36.535823	13.631860	47569.633384	
Tennessee	-17.435007	-31.860414	10.611521	16029.597489	
North Carolina	-25.759184	-32.126077	11.143764	21065.411338	
Minnesota	-31.470244	-36.924878	13.820488	13669.610732	
Nebraska	-9.994106	-26.013752	9.622790	6951.964637	
Alabama	-11.271468	-29.243767	10.091413	14373.312096	
Michigan	-29.829510	-42.265964	15.245505	38391.647241	
Texas	-16.808856	-28.565618	11.000801	51283.818473	
Louisiana	-25.594109	-31.798132	11.641523	30850.614943	
Florida	-29.198368	-29.913053	10.696436	53416.045513	

Connecticut	-36.258242	-36.258242	14.302198	25056.899725
Nevada	-40.930348	-38.758706	12.711443	7268.263682
Massachusetts	-33.793713	-34.920760	13.502292	53715.744597
Arizona	-14.281912	-26.998790	8.793708	20768.525711
Illinois	-22.368318	-31.233938	11.388916	59435.843060
New Jersey	-39.093836	-40.564857	16.826587	99641.728611
District of Columbia	-65.574803	-52.771654	19.661417	5572.606299
New York	-39.001820	-38.962518	15.473071	238659.528384

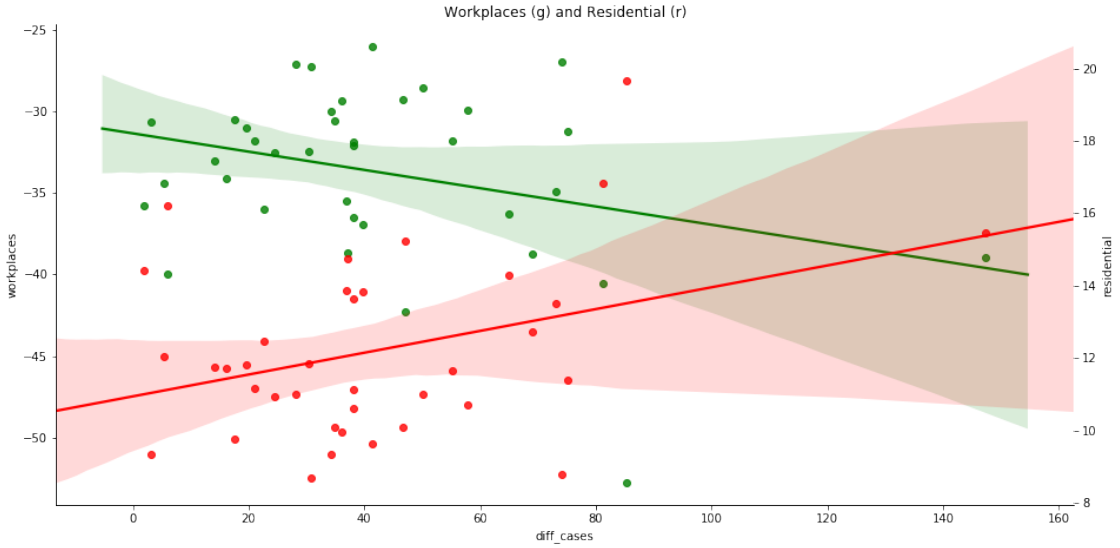
state	deaths	restaurant_performance	diff_cases
Hawaii	11.943615	-89.583682	1.990050
Oregon	101.717728	-78.748290	3.090075
Ohio	1201.014916	-80.849797	5.431981
Maryland	1353.592278	-85.688278	5.938996
New Mexico	229.229692	-83.717535	14.028011
Kentucky	260.202322	-77.959386	16.064677
Washington	654.699170	-76.814282	17.502075
California	2462.528164	-74.256525	19.574982
Georgia	1260.059747	-75.705648	21.054514
Colorado	855.841523	-77.136990	22.684889
Missouri	452.037879	-79.733510	24.555556
Wisconsin	268.136596	-67.693185	28.059167
Indiana	1040.900000	-77.659947	30.416842
Oklahoma	206.556904	-66.266586	30.704097
Utah	65.150289	-69.470358	34.163006
Kansas	128.623853	-75.168917	34.829358
South Carolina	330.473617	-71.261850	36.011443
Rhode Island	334.104408	-65.821879	36.798144
Virginia	786.940063	-83.502871	37.155100
Pennsylvania	3147.080030	-85.528800	38.030488
Tennessee	244.460118	-76.013360	38.041359
North Carolina	544.836735	-80.622594	38.134240
Minnesota	591.915122	-78.661776	39.808780
Nebraska	95.188605	-63.397250	41.306483
Alabama	424.285319	-75.112308	46.631579
Michigan	3351.249845	-84.992393	47.035958
Texas	994.051693	-65.177678	50.079343
Louisiana	1862.991379	-76.974784	55.251437
Florida	1580.261700	-70.125447	57.922070
Connecticut	2174.004121	-77.715412	64.953297
Nevada	267.134328	-77.806692	69.012438
Massachusetts	3587.176162	-74.128134	73.081860
Arizona	577.629764	-63.913618	74.094374
Illinois	2748.536538	-75.359044	75.234919
New Jersey	6966.889604	-82.800133	81.220791
District of Columbia	284.732283	-90.731654	85.401575

```
New York              17957.172489              -86.447260  147.378457
```

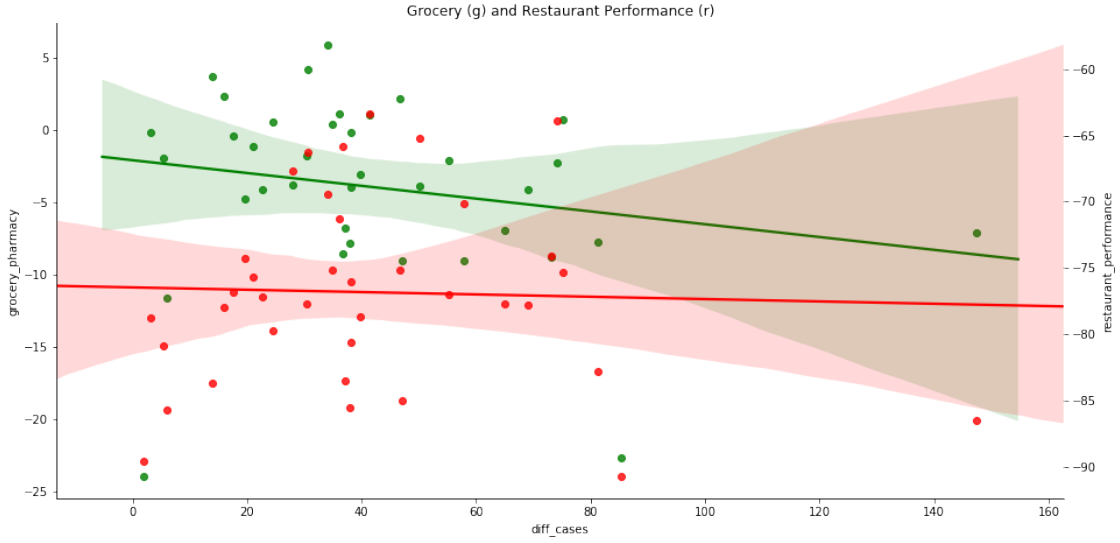
[86]:
```
# Plot the relationship between number of cases and parks and retail/recreation␣
 ↪mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(state['diff_cases'], state['parks'], ax=ax, color='g')
ax2 = ax.twinx()
sns.regplot(state['diff_cases'], state['retail_recreation'], ax=ax2, color='r').
 ↪set_title('Parks (g) and Retail/Recreation (r)')
sns.despine()
```



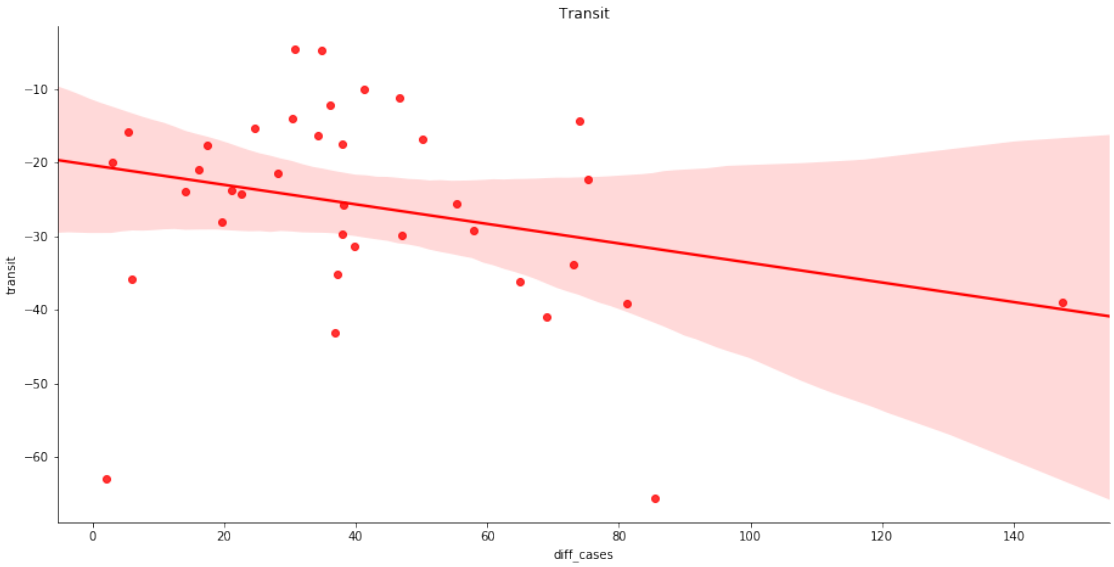
[87]:
```
# Plot the relationship between number of cases and workplaces and residential␣
 ↪mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(state['diff_cases'], state['workplaces'], ax=ax, color='g')
ax2 = ax.twinx()
sns.regplot(state['diff_cases'], state['residential'], ax=ax2, color='r').
 ↪set_title('Workplaces (g) and Residential (r)')
sns.despine()
```



[88]:
```
# Plot the relationship between number of cases and parks and retail/recreation␣
↪mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(state['diff_cases'], state['grocery_pharmacy'], ax=ax, color='g')
ax2 = ax.twinx()
sns.regplot(state['diff_cases'], state['restaurant_performance'], ax=ax2,␣
↪color='r').set_title('Grocery (g) and Restaurant Performance (r)')
sns.despine()
```



[89]:
```
# Plot the relationship between number of cases and parks and retail/recreation␣
 ↪mobility
fig, ax = plt.subplots()
fig.set_size_inches(15.5, 7.5)
sns.regplot(state['diff_cases'], state['transit'], ax=ax, color='r').
 ↪set_title('Transit')
sns.despine()
```



[90]:
```
# Create new dataframe from the percent change in mobility in each area and␣
 ↪group by state
change = all_data_new.sort_index(level=['state', 'date'])

change['retail_recreation'] = change['retail_recreation'].astype(int)
change['grocery_pharmacy'] = change['grocery_pharmacy'].astype(int)
change['parks'] = change['parks'].astype(int)
change['transit'] = change['transit'].astype(int)
change['workplaces'] = change['workplaces'].astype(int)
change['residential'] = change['residential'].astype(int)

change['pct_ch_rec'] = (change.groupby('state')['retail_recreation'].apply(pd.
 ↪Series.pct_change))
change['pct_ch_groc'] = (change.groupby('state')['grocery_pharmacy'].apply(pd.
 ↪Series.pct_change))
change['pct_ch_parks'] = (change.groupby('state')['parks'].apply(pd.Series.
 ↪pct_change))
change['pct_ch_transit'] = (change.groupby('state')['transit'].apply(pd.Series.
 ↪pct_change))
```

```
change['pct_ch_work'] = (change.groupby('state')['workplaces'].apply(pd.Series.
 ↪pct_change))
change['pct_ch_residential'] = (change.groupby('state')['residential'].apply(pd.
 ↪Series.pct_change))
change['pct_ch_rest'] = (change.groupby('state')['restaurant_performance'].
 ↪apply(pd.Series.pct_change))
change['pct_ch_cases'] = (change.groupby('state')['cases'].apply(pd.Series.
 ↪pct_change))

#drop Nan values
change = change.replace([np.inf, -np.inf], np.nan)
change = change.dropna()

date_norm = change.groupby(['state']).mean()

date_norm = date_norm.drop(['cases', 'deaths', 'restaurant_performance'], axis=1)

date_norm  = date_norm.sort_values(['pct_ch_cases'])

date_norm
```

[90]:

state	retail_recreation	grocery_pharmacy	parks	\
Hawaii	-44.913194	-25.180556	-50.798611	
District of Columbia	-56.545455	-24.107438	-43.214876	
Nebraska	-17.509934	0.865342	74.982340	
Washington	-21.419308	-1.186840	54.231988	
California	-33.529277	-5.847051	3.734945	
Oregon	-21.124679	-0.930591	49.273136	
Illinois	-26.220705	-0.501138	44.436291	
Arizona	-19.544077	-3.375344	0.398760	
Massachusetts	-28.615958	-10.283371	61.702461	
Colorado	-25.067559	-4.982609	42.972575	
Utah	-16.737052	4.604250	80.585657	
Florida	-27.383874	-10.498590	-16.216502	
Oklahoma	-13.050505	4.109428	28.671717	
Alabama	-19.073120	1.765191	18.811535	
Texas	-19.607365	-4.881243	6.223015	
Kansas	-19.746939	0.177551	70.444898	
Ohio	-24.900000	-2.828859	79.653020	
Maryland	-30.093388	-12.579339	53.600000	
Georgia	-20.107178	-2.080629	17.010324	
Pennsylvania	-30.607038	-8.988689	53.635945	
Michigan	-33.276768	-10.138721	87.964983	
New York	-40.052505	-8.290982	51.757916	
Connecticut	-26.475556	-8.029630	48.437037	
Rhode Island	-25.517327	-9.111386	70.589109	

Louisiana	-25.308544	-2.818038	-5.564873
Minnesota	-28.383333	-3.704167	82.304167
South Carolina	-16.625529	0.703808	20.309591
North Carolina	-23.675635	-4.754315	25.586802
New Mexico	-22.713147	3.007968	11.075697
Tennessee	-19.143568	-0.641494	27.774274
Wisconsin	-28.264860	-5.388112	71.523601
Kentucky	-23.131627	1.873857	53.287020
Virginia	-29.039548	-7.442938	29.871751
Indiana	-23.948441	-2.919664	49.382494
New Jersey	-39.894789	-8.555556	51.334317
Missouri	-21.953008	0.061090	47.567669
Nevada	-27.278820	-4.900804	-7.316354

state	transit	workplaces	residential	pct_ch_rec	\
Hawaii	-65.626736	-37.362847	15.041667	0.028645	
District of Columbia	-68.595041	-55.157025	20.553719	0.018593	
Nebraska	-10.708609	-27.059603	10.064018	0.045233	
Washington	-20.239193	-33.161864	10.692123	0.071370	
California	-31.057929	-34.087727	13.008127	0.034879	
Oregon	-21.830334	-32.569409	9.977506	0.003022	
Illinois	-25.232082	-34.479522	12.618316	0.064178	
Arizona	-16.654270	-29.425620	9.642562	-0.045870	
Massachusetts	-37.565250	-38.413870	14.894109	0.018180	
Colorado	-25.583946	-37.175920	12.957860	-0.005443	
Utah	-18.237716	-32.135458	10.135458	-0.057919	
Florida	-31.772685	-31.987071	11.455571	-0.018162	
Oklahoma	-4.984848	-28.146465	8.962963	-0.084601	
Alabama	-11.988671	-29.873326	10.331617	-0.023984	
Texas	-18.958803	-30.967089	11.993556	-0.003944	
Kansas	-5.116327	-31.269388	10.442857	-0.078227	
Ohio	-16.785906	-35.571141	12.561745	-0.019064	
Maryland	-37.649587	-41.797521	16.955372	0.022111	
Georgia	-25.828417	-33.354966	11.771878	-0.016416	
Pennsylvania	-31.801005	-38.277336	14.379975	-0.049029	
Michigan	-31.311785	-43.559596	15.787205	0.016137	
New York	-41.645691	-41.645691	16.511022	-0.069551	
Connecticut	-37.967407	-37.737778	14.914074	-0.020301	
Rhode Island	-44.428218	-36.586634	14.349010	0.054938	
Louisiana	-26.716772	-32.588608	11.978639	0.032455	
Minnesota	-32.534375	-37.959375	14.290625	0.037030	
South Carolina	-12.958392	-30.160085	10.264457	-0.042768	
North Carolina	-27.561929	-33.853299	11.785279	0.016911	
New Mexico	-24.728088	-33.351594	11.907371	-0.050360	
Tennessee	-17.926141	-32.812448	10.990871	0.028765	
Wisconsin	-25.338287	-31.439685	12.701049	0.066122	

Kentucky	-21.723949	-35.277879	12.113346	0.024242
Virginia	-36.660452	-39.821469	15.264407	0.034253
Indiana	-15.899281	-34.333333	12.642686	-0.019088
New Jersey	-40.708948	-42.064405	17.455752	-0.020358
Missouri	-16.661654	-33.295113	11.309211	-0.130181
Nevada	-42.659517	-40.402145	13.278820	-0.039717

state	pct_ch_groc	pct_ch_parks	pct_ch_transit	pct_ch_work	\
Hawaii	0.023504	0.082991	-0.017249	0.035083	
District of Columbia	0.052138	0.386738	0.015155	0.080710	
Nebraska	-0.568611	-0.319389	-0.078857	0.135880	
Washington	-0.102238	0.018537	-0.080737	0.045408	
California	-0.015338	-0.130100	-0.066874	0.085459	
Oregon	-0.006196	-0.225703	0.039258	0.041260	
Illinois	-0.263651	-0.446133	-0.081791	0.092537	
Arizona	0.028968	-0.025976	-0.025499	0.013811	
Massachusetts	-0.066325	-0.497316	-0.129188	0.095195	
Colorado	-0.099917	-0.073126	0.090947	0.110478	
Utah	0.140715	-0.015543	-0.187163	0.098710	
Florida	0.002238	-0.106240	-0.028965	0.063093	
Oklahoma	-0.103508	-0.559733	-0.092520	0.101098	
Alabama	-0.247503	-0.213500	-0.011964	0.112412	
Texas	-0.091175	-0.178523	-0.064659	0.047318	
Kansas	-0.172614	-0.046237	-0.022564	0.087497	
Ohio	-0.004567	0.083395	-0.091409	0.096781	
Maryland	-0.001936	0.226274	-0.045566	0.135849	
Georgia	-0.165488	-0.209467	-0.022538	0.061770	
Pennsylvania	-0.017130	-0.481883	-0.004511	0.157645	
Michigan	-0.135281	-0.090882	-0.039082	0.098083	
New York	-0.142688	-0.367505	0.005074	0.097382	
Connecticut	-0.024821	-0.224094	0.008356	0.159295	
Rhode Island	-0.096747	-0.825220	-0.037609	0.172253	
Louisiana	-0.211509	-0.236437	-0.085397	0.093898	
Minnesota	0.074770	-0.237945	-0.091530	0.112880	
South Carolina	-0.135133	-0.305699	-0.075113	0.060026	
North Carolina	-0.164716	-0.322825	-0.041076	0.065264	
New Mexico	-0.063879	-0.014625	-0.022284	0.089784	
Tennessee	-0.272149	-0.092023	-0.029986	0.125056	
Wisconsin	-0.124206	-0.054082	-0.058650	0.207395	
Kentucky	-0.073098	-0.221476	0.040647	0.129124	
Virginia	-0.145252	-0.297847	-0.031031	0.120668	
Indiana	-0.197184	0.004487	-0.093143	0.078505	
New Jersey	-0.097923	-0.311685	-0.008664	0.181096	
Missouri	-0.279925	-0.285069	-0.023425	0.113334	
Nevada	0.007721	0.097580	-0.005903	0.035738	

state	pct_ch_residential	pct_ch_rest	pct_ch_cases
Hawaii	0.048946	0.013587	0.061123
District of Columbia	0.120525	0.006007	0.064354
Nebraska	0.131469	-1.018203	0.064665
Washington	0.105404	-0.057370	0.075155
California	0.058904	-0.005226	0.076138
Oregon	0.067272	-0.002264	0.081859
Illinois	0.169491	-0.137003	0.089177
Arizona	0.032001	-0.002061	0.089547
Massachusetts	0.115392	-0.242940	0.102691
Colorado	0.111448	0.029759	0.103330
Utah	0.081129	0.042371	0.107338
Florida	0.041515	0.036428	0.110736
Oklahoma	0.175397	-0.221443	0.113196
Alabama	0.175739	0.044123	0.113224
Texas	0.062946	0.020575	0.116881
Kansas	0.153807	0.287851	0.118709
Ohio	0.120032	-0.043038	0.119495
Maryland	0.083923	0.042386	0.120756
Georgia	0.099055	-0.020833	0.122403
Pennsylvania	0.165202	-0.019266	0.123894
Michigan	0.127387	0.042923	0.125589
New York	0.047874	0.046482	0.128998
Connecticut	0.206992	0.033762	0.130754
Rhode Island	0.106462	-0.053828	0.131685
Louisiana	0.140125	0.004744	0.137449
Minnesota	0.057686	-0.015312	0.157416
South Carolina	0.126625	-0.001795	0.166674
North Carolina	0.128322	0.061696	0.170965
New Mexico	0.117196	-0.019930	0.174214
Tennessee	0.148653	0.031218	0.186123
Wisconsin	-0.017340	-0.075552	0.229824
Kentucky	0.180372	0.028959	0.298353
Virginia	0.133905	-0.000160	0.306226
Indiana	0.150465	-0.166191	0.317455
New Jersey	0.105498	0.031144	0.358670
Missouri	0.131293	0.036396	0.545830
Nevada	0.050368	0.043686	0.902639

3 Airports/International Travel and Stocks

3.0.1 Airport Traffic

[91]:
```
traffic = pd.read_csv('data/airlines_travel/airport-traffic.csv', sep= ',',␣
 ↪header= None)

# a summary of the airport traffic file
traffic.head()
```

[91]:
```
                 0           1        2                                3   \
0  aggregationmethod        date  version                      airportname
1              Daily  2020-03-31        1       Boston Logan International
2              Daily  2020-03-31        1            Calgary International
3              Daily  2020-03-31        1  Charlotte Douglas International
4              Daily  2020-03-31        1      Chicago OHare International

                  4                                     5          6   \
0  percentofbaseline                              centroid       city
1                 71  POINT (-71.0102909977 42.3636330377)     Boston
2                 91  POINT (-114.013122872 51.1184753728)    Calgary
3                 81  POINT (-80.9478114283 35.2136892261)  Charlotte
4                 77   POINT (-87.910595204 41.9804600429)    Chicago

              7          8                             9   \
0         state  iso_3166_2                       country
1   Massachusetts       US-MA  United States of America (the)
2         Alberta       CA-AB                        Canada
3  North Carolina       US-NC  United States of America (the)
4        Illinois       US-IL  United States of America (the)

                                           10
0                                   geography
1  POLYGON ((-71.005089283 42.3472534333, -71.003...
2  POLYGON ((-113.981866837 51.1392131914, -113.9...
3  POLYGON ((-80.9332966805 35.2337368341, -80.94...
4  POLYGON ((-87.9397845268 41.9608463705, -87.93...
```

[92]:
```
new_header = traffic.iloc[0]
traffic = traffic[1:]
traffic.columns = new_header
traffic.head()
```

[92]:
```
0 aggregationmethod        date version                      airportname  \
1              Daily  2020-03-31       1       Boston Logan International
2              Daily  2020-03-31       1            Calgary International
3              Daily  2020-03-31       1  Charlotte Douglas International
4              Daily  2020-03-31       1      Chicago OHare International
5              Daily  2020-03-31       1  Dallas/Fort Worth International

0 percentofbaseline                              centroid       city  \
```

```
1               71  POINT (-71.0102909977 42.3636330377)     Boston
2               91  POINT (-114.013122872 51.1184753728)    Calgary
3               81  POINT (-80.9478114283 35.2136892261)  Charlotte
4               77   POINT (-87.910595204 41.9804600429)    Chicago
5               68  POINT (-97.0394983969 32.8940590356)  Grapevine

0          state iso_3166_2                         country  \
1   Massachusetts      US-MA  United States of America (the)
2         Alberta      CA-AB                          Canada
3  North Carolina      US-NC  United States of America (the)
4        Illinois      US-IL  United States of America (the)
5           Texas      US-TX  United States of America (the)

0                                          geography
1  POLYGON ((-71.005089283 42.3472534333, -71.003...
2  POLYGON ((-113.981866837 51.1392131914, -113.9...
3  POLYGON ((-80.9332966805 35.2337368341, -80.94...
4  POLYGON ((-87.9397845268 41.9608463705, -87.93...
5  POLYGON ((-97.0429444313 32.9265900261, -97.04...
```

[93]:
```
traffic.shape
```

[93]:
```
(50, 11)
```

[94]:
```
traffic.describe(include = 'all')
```

[94]:
```
0      aggregationmethod        date version       airportname  \
count                 50          50      50                50
unique                 1           2       1                26
top                Daily  2020-03-31       1  Montreal Trudeau
freq                  50          26      50                 2

0      percentofbaseline                                centroid      city  \
count                 50                                     50        50
unique                33                                     26        25
top                   57  POINT (-104.700315559 39.8643468206)  New York
freq                   3                                      2         4

0         state iso_3166_2                         country  \
count        50         50                              50
unique       21         21                               2
top     Alberta      CA-ON  United States of America (the)
freq          4          4                              33

0                                           geography
count                                              50
unique                                             26
```

```
top     POLYGON ((-123.136525154 49.1980971491, -123.1...
freq                                                    2
```

[95]:
```
traffic["airportname"].value_counts()
```

[95]:
```
Montreal Trudeau                          2
Seattle-Tacoma International              2
Detroit Metropolitan Wayne County         2
Vancouver International                   2
San Francisco International               2
Edmonton International                    2
Hamilton International                    2
Boston Logan International                2
Miami International                       2
Calgary International                     2
LaGuardia                                 2
John F. Kennedy International             2
Dallas/Fort Worth International           2
Los Angeles International                 2
Denver International                      2
Hartsfield-Jackson Atlanta International  2
Charlotte Douglas International           2
Montreal Mirabel                          2
Newark Liberty International              2
McCarran International                    2
Chicago OHare International               2
Toronto Pearson                           2
Halifax International                     2
Daniel K. Inouye International            2
Washington Dulles International           1
Winnipeg International                    1
Name: airportname, dtype: int64
```

[96]:
```
traffic["date"].value_counts()
```

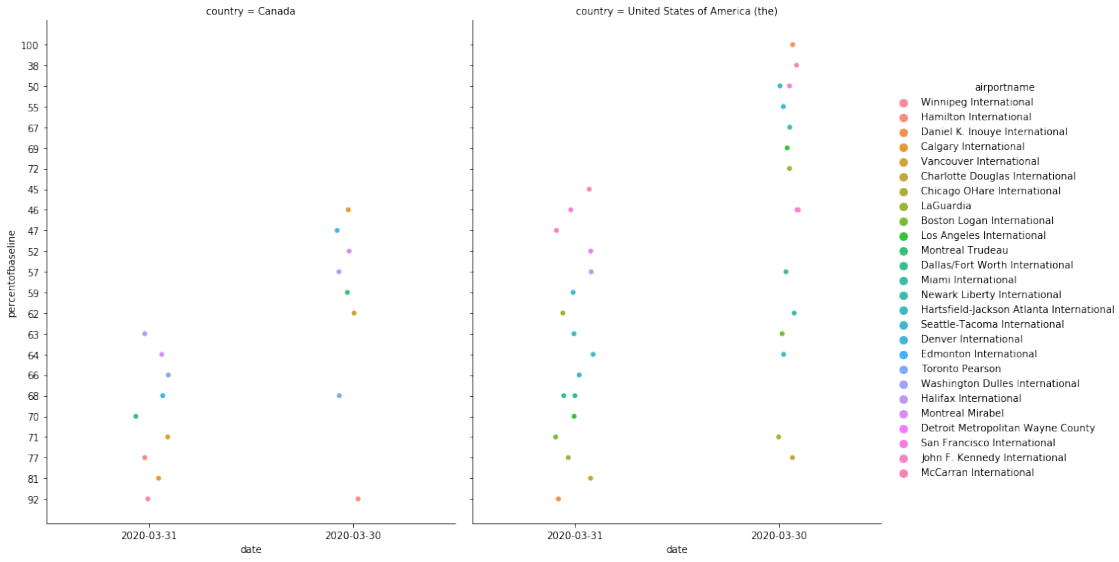
[96]:
```
2020-03-31    26
2020-03-30    24
Name: date, dtype: int64
```

[97]:
```
taffic = traffic.sort_values(by=['percentofbaseline'], inplace=True,␣
 ↪ascending=False)
traffic.head()
```

[97]:
```
0  aggregationmethod        date version                    airportname  \
26             Daily  2020-03-31       1          Winnipeg International
37             Daily  2020-03-30       1          Hamilton International
6              Daily  2020-03-31       1  Daniel K. Inouye International
```

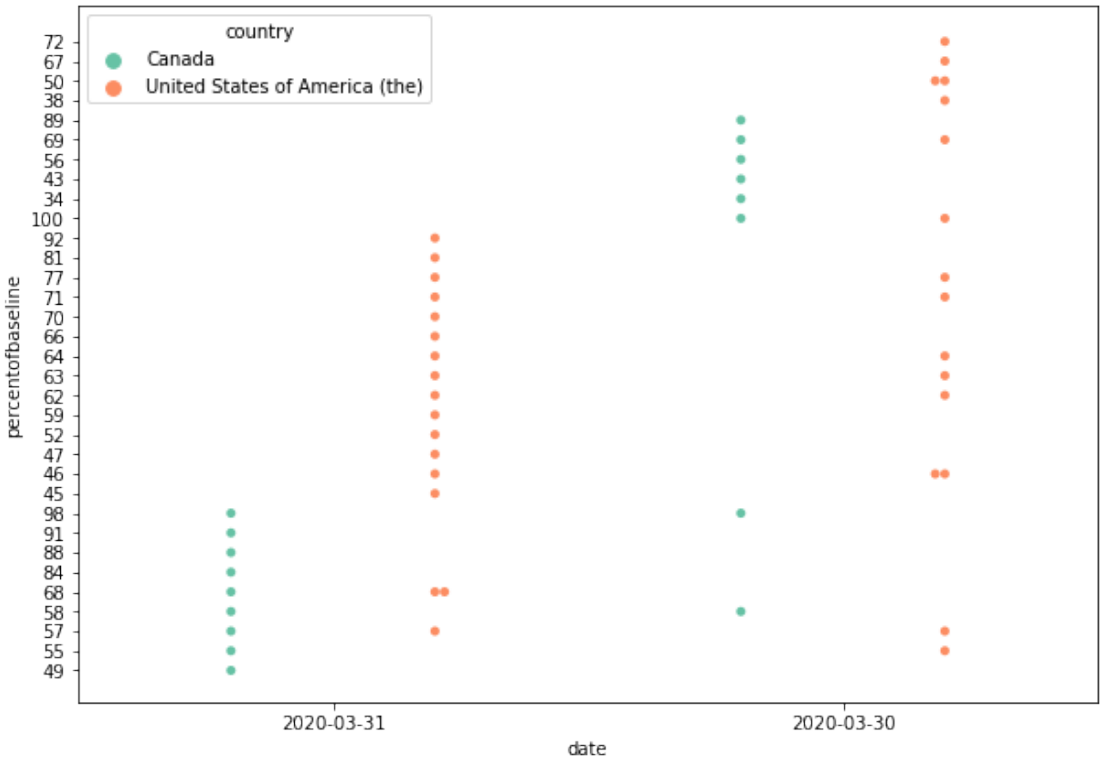
```
2              Daily  2020-03-31       1           Calgary International
50             Daily  2020-03-30       1         Vancouver International

0  percentofbaseline                                centroid           city  \
26                98  POINT (-97.2190621862 49.9024712566)        Winnipeg
37                98  POINT (-79.9266930702 43.1720360845)        Hamilton
6                 92  POINT (-157.918285205 21.3259652489)  Urban Honolulu
2                 91  POINT (-114.013122872 51.1184753728)         Calgary
50                89  POINT (-123.177541243 49.1935788601)        Richmond

0              state iso_3166_2                         country  \
26          Manitoba      CA-MB                          Canada
37           Ontario      CA-ON                          Canada
6             Hawaii      US-HI  United States of America (the)
2            Alberta      CA-AB                          Canada
50  British Columbia      CA-BC                          Canada

0                                          geography
26  POLYGON ((-97.2094345093 49.8979517851, -97.20...
37  POLYGON ((-79.9327468872 43.1551600162, -79.93...
6   POLYGON ((-157.926621437 21.3389395548, -157.9...
2   POLYGON ((-113.981866837 51.1392131914, -113.9...
50  POLYGON ((-123.136525154 49.1980971491, -123.1...
```

[98]:
```
ax = sns.catplot(x="date", y="percentofbaseline", hue="airportname",␣
↪col="country", data=traffic,
                 height=8, aspect=.8)
```

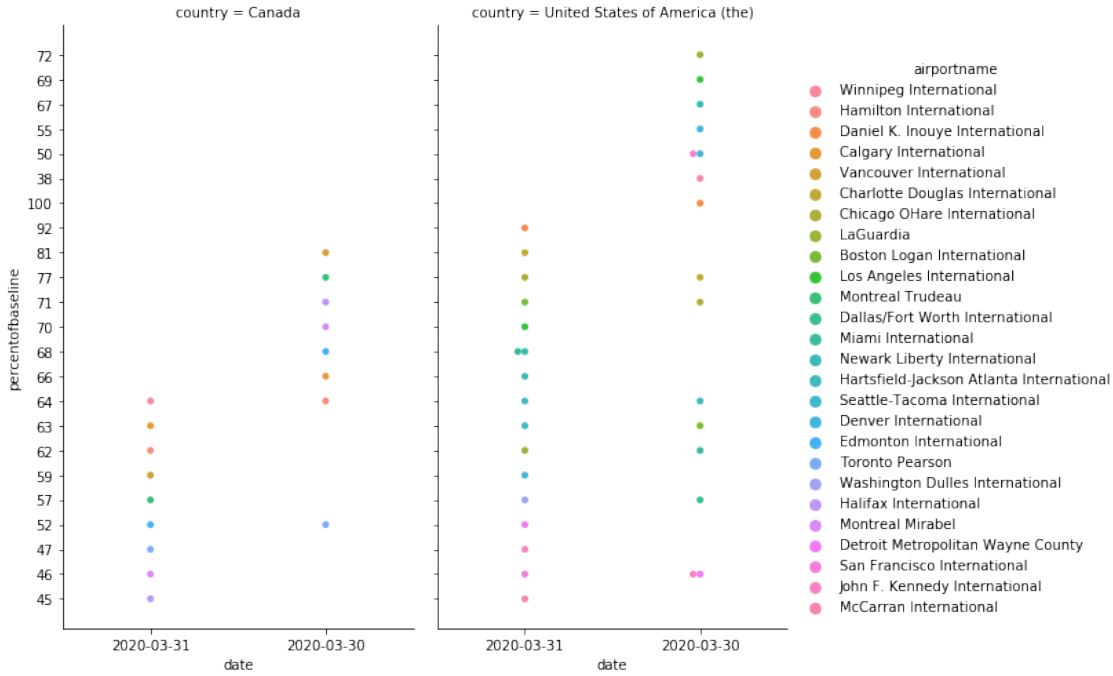


[99]:
```
fig_dims = (10, 7)
fig, ax = plt.subplots(figsize=fig_dims)
ax = sns.swarmplot(x="date", y="percentofbaseline", hue="country",
                   data=traffic, palette="Set2", dodge=True)
```

```
C:\Users\nic_v\Anaconda3\lib\site-packages\seaborn\categorical.py:1243:
RuntimeWarning: invalid value encountered in sqrt
  dx = np.sqrt(d ** 2 - dy ** 2) * 1.05
```



[100]:
```
ax = sns.catplot(x="date", y="percentofbaseline",
                 hue="airportname", col="country",
                 data=traffic, kind="swarm",
                 height=7, aspect=.6);
```



3.0.2 Airport Stock Data

American Airlines

[101]:
```
american_air = pd.read_csv('data/airlines_travel/AAL.csv', sep= ',', header=␣
↪None)

# a summary of the travel restrictions file
american_air.head()
```

[101]:

	0	1	2	3	4	5	6
0	Date	Open	High	Low	Close	Adj Close	Volume
1	2019-06-26	31.480000	31.719999	31.070000	31.459999	31.133385	4089700
2	2019-06-27	31.559999	32.549999	31.420000	32.150002	31.816221	4274500
3	2019-06-28	32.230000	32.730000	32.160000	32.610001	32.271446	5238500
4	2019-07-01	33.139999	33.660000	32.529999	32.880001	32.538643	8985600

[102]:
```
new_header = american_air.iloc[0]
american_air = american_air[1:]
american_air.columns = new_header
american_air.head()
```

[102]:

0	Date	Open	High	Low	Close	Adj Close	Volume
1	2019-06-26	31.480000	31.719999	31.070000	31.459999	31.133385	4089700
2	2019-06-27	31.559999	32.549999	31.420000	32.150002	31.816221	4274500

```
3  2019-06-28  32.230000  32.730000  32.160000  32.610001  32.271446  5238500
4  2019-07-01  33.139999  33.660000  32.529999  32.880001  32.538643  8985600
5  2019-07-02  33.090000  33.209999  32.029999  32.189999  31.855806  4765100
```

[103]:
```
american_air.dtypes
```

[103]:
```
0
Date         object
Open         object
High         object
Low          object
Close        object
Adj Close    object
Volume       object
dtype: object
```

[104]:
```
american_air["Volume"] = american_air["Volume"].astype(str).astype(int)
american_air["Open"] = american_air["Open"].astype(str).astype(float)
american_air["High"] = american_air["High"].astype(str).astype(float)
american_air["Low"] = american_air["Low"].astype(str).astype(float)
american_air["Close"] = american_air["Close"].astype(str).astype(float)
american_air.dtypes
```

[104]:
```
0
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close     object
Volume         int32
dtype: object
```

[105]:
```
american_air = american_air[~american_air.Date.str.contains('2019')]
american_air['Month'] = pd.DatetimeIndex(american_air['Date']).month
american_air['Day'] = pd.DatetimeIndex(american_air['Date']).day
american_air.head()
```

[105]:
```
0          Date       Open       High        Low      Close  Adj Close  \
132  2020-01-02  28.980000  29.299999  28.650000  29.090000  28.982893
133  2020-01-03  28.270000  28.290001  27.340000  27.650000  27.548195
134  2020-01-06  27.190001  27.490000  27.080000  27.320000  27.219410
135  2020-01-07  27.559999  27.680000  27.059999  27.219999  27.119778
136  2020-01-08  27.100000  28.090000  27.070000  27.840000  27.737495

0      Volume  Month  Day
132   6451100      1    2
```

```
133  14008900      1    3
134   6105800      1    6
135   6105900      1    7
136  10496800      1    8
```

Delta Airlines

[106]:
```
delta = pd.read_csv('data/airlines_travel/DAL.csv', sep= ',', header= None)

# a summary of the travel restrictions file
delta.head()
```

[106]:

	0	1	2	3	4	5	6
0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	42.110001	42.490002	41.799999	42.169998	38.448864	10701800
2	2016-05-03	42.919998	43.200001	41.830002	42.919998	39.132687	14319500
3	2016-05-04	42.369999	42.560001	41.169998	41.430000	37.774166	14925300
4	2016-05-05	41.590000	42.270000	41.450001	41.750000	38.065926	10271100

[107]:
```
new_header = delta.iloc[0]
delta = delta[1:]
delta.columns = new_header
delta.head()
```

[107]:

0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	42.110001	42.490002	41.799999	42.169998	38.448864	10701800
2	2016-05-03	42.919998	43.200001	41.830002	42.919998	39.132687	14319500
3	2016-05-04	42.369999	42.560001	41.169998	41.430000	37.774166	14925300
4	2016-05-05	41.590000	42.270000	41.450001	41.750000	38.065926	10271100
5	2016-05-06	41.610001	42.259998	40.950001	42.040001	38.330345	10777200

[108]:
```
delta["Volume"] = delta["Volume"].astype(str).astype(int)
delta["Open"] = delta["Open"].astype(str).astype(float)
delta["High"] = delta["High"].astype(str).astype(float)
delta["Low"] = delta["Low"].astype(str).astype(float)
delta["Close"] = delta["Close"].astype(str).astype(float)
delta.dtypes
```

[108]:
```
0
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close     object
Volume         int32
dtype: object
```

[109]:
```
delta = delta[~delta.Date.str.contains('2016')]
delta = delta[~delta.Date.str.contains('2017')]
delta = delta[~delta.Date.str.contains('2018')]
delta = delta[~delta.Date.str.contains('2019')]
delta['Month'] = pd.DatetimeIndex(delta['Date']).month
delta['Day'] = pd.DatetimeIndex(delta['Date']).day
delta.head()
```

[109]:

0	Date	Open	High	Low	Close	Adj Close	\
925	2020-01-02	58.930000	59.389999	58.450001	59.040001	58.635307	
926	2020-01-03	57.500000	58.119999	56.910000	58.060001	57.662025	
927	2020-01-06	56.990002	57.759998	56.660000	57.660000	57.264767	
928	2020-01-07	57.910000	58.070000	57.470001	57.610001	57.215111	
929	2020-01-08	57.750000	59.400002	57.730000	58.849998	58.446609	

0	Volume	Month	Day
925	4459200	1	2
926	9078100	1	3
927	5504300	1	6
928	5563000	1	7
929	8519000	1	8

Southwest Airlines

[110]:
```
southwest = pd.read_csv('data/airlines_travel/LUV.csv', sep= ',', header= None)

# a summary of the travel restrictions file
southwest.head()
```

[110]:

	0	1	2	3	4	5	6
0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	44.630001	45.000000	43.930000	44.009998	42.149231	5482800
2	2016-05-03	44.230000	44.279999	42.939999	43.139999	41.316013	6220500
3	2016-05-04	42.799999	42.810001	41.389999	41.660000	39.898590	7609600
4	2016-05-05	41.759998	42.139999	41.389999	41.529999	39.774086	5969100

[111]:
```
new_header = southwest.iloc[0]
southwest = southwest[1:]
southwest.columns = new_header
southwest.head()
```

[111]:

0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	44.630001	45.000000	43.930000	44.009998	42.149231	5482800
2	2016-05-03	44.230000	44.279999	42.939999	43.139999	41.316013	6220500
3	2016-05-04	42.799999	42.810001	41.389999	41.660000	39.898590	7609600
4	2016-05-05	41.759998	42.139999	41.389999	41.529999	39.774086	5969100
5	2016-05-06	41.470001	41.939999	40.740002	41.730000	39.965622	6550400

[112]:
```
southwest["Volume"] = southwest["Volume"].astype(str).astype(int)
southwest["Open"] = southwest["Open"].astype(str).astype(float)
southwest["High"] = southwest["High"].astype(str).astype(float)
southwest["Low"] = southwest["Low"].astype(str).astype(float)
southwest["Close"] = southwest["Close"].astype(str).astype(float)
southwest.dtypes
```

[112]:
```
0
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close     object
Volume         int32
dtype: object
```

[113]:
```
southwest = southwest[~southwest.Date.str.contains('2016')]
southwest = southwest[~southwest.Date.str.contains('2017')]
southwest = southwest[~southwest.Date.str.contains('2018')]
southwest = southwest[~southwest.Date.str.contains('2019')]
southwest['Month'] = pd.DatetimeIndex(southwest['Date']).month
southwest['Day'] = pd.DatetimeIndex(southwest['Date']).day
southwest.head()
```

[113]:

0	Date	Open	High	Low	Close	Adj Close	Volume	Month	Day
925	2020-01-02	54.380001	54.869999	54.290001	54.840000	54.629749	3713000	1	2
926	2020-01-03	53.599998	54.419998	53.040001	54.349998	54.141628	3536100	1	3
927	2020-01-06	53.689999	54.560001	53.480000	54.130001	53.922474	4200300	1	6
928	2020-01-07	54.330002	54.400002	53.880001	54.290001	54.081860	2997600	1	7
929	2020-01-08	53.680000	54.660000	53.639999	54.369999	54.161549	4103100	1	8

Spirit Airlines

[114]:
```
spirit_air = pd.read_csv('data/airlines_travel/SAVE.csv', sep= ',', header= None)

# a summary of the travel restrictions file
spirit_air.head()
```

[114]:

	0	1	2	3	4	5	6
0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	45.480000	45.680000	43.570000	43.820000	43.820000	1511900
2	2016-05-03	43.580002	43.990002	42.849998	43.150002	43.150002	1423400
3	2016-05-04	42.730000	43.110001	41.299999	41.349998	41.349998	1190000
4	2016-05-05	41.950001	43.139999	41.450001	41.889999	41.889999	1035400

[115]:
```
new_header = spirit_air.iloc[0]
spirit_air = spirit_air[1:]
spirit_air.columns = new_header
spirit_air.head()
```

[115]:

0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	45.480000	45.680000	43.570000	43.820000	43.820000	1511900
2	2016-05-03	43.580002	43.990002	42.849998	43.150002	43.150002	1423400
3	2016-05-04	42.730000	43.110001	41.299999	41.349998	41.349998	1190000
4	2016-05-05	41.950001	43.139999	41.450001	41.889999	41.889999	1035400
5	2016-05-06	41.360001	41.919998	40.480000	41.080002	41.080002	1141000

[116]:
```
spirit_air["Volume"] = spirit_air["Volume"].astype(str).astype(int)
spirit_air["Open"] = spirit_air["Open"].astype(str).astype(float)
spirit_air["High"] = spirit_air["High"].astype(str).astype(float)
spirit_air["Low"] = spirit_air["Low"].astype(str).astype(float)
spirit_air["Close"] = spirit_air["Close"].astype(str).astype(float)
spirit_air.dtypes
```

[116]:
```
0
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close     object
Volume         int32
dtype: object
```

[117]:
```
spirit_air = spirit_air[~spirit_air.Date.str.contains('2016')]
spirit_air = spirit_air[~spirit_air.Date.str.contains('2017')]
spirit_air = spirit_air[~spirit_air.Date.str.contains('2018')]
spirit_air = spirit_air[~spirit_air.Date.str.contains('2019')]
spirit_air['Month'] = pd.DatetimeIndex(spirit_air['Date']).month
spirit_air['Day'] = pd.DatetimeIndex(spirit_air['Date']).day
spirit_air.head()
```

[117]:

0	Date	Open	High	Low	Close	Adj Close	\
925	2020-01-02	40.700001	40.930000	40.099998	40.650002	40.650002	
926	2020-01-03	39.660000	39.930000	38.330002	39.820000	39.820000	

```
927  2020-01-06  39.299999  40.540001  39.049999  40.389999  40.389999
928  2020-01-07  40.220001  40.380001  39.220001  39.500000  39.500000
929  2020-01-08  39.119999  40.540001  39.009998  39.939999  39.939999

0      Volume  Month  Day
925  1027400      1    2
926  1327700      1    3
927  1003500      1    6
928   866700      1    7
929  1193700      1    8
```

United Airlines

[118]:
```
united = pd.read_csv('data/airlines_travel/UAL.csv', sep= ',', header= None)

# a summary of the travel restrictions file
united.head()
```

[118]:

	0	1	2	3	4	5	6
0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	46.610001	46.810001	45.810001	46.660000	46.660000	6626200
2	2016-05-03	47.000000	47.540001	46.150002	47.130001	47.130001	6726600
3	2016-05-04	46.709999	46.759998	45.150002	45.549999	45.549999	7184800
4	2016-05-05	45.599998	46.070000	44.959999	44.990002	44.990002	6037400

[119]:
```
new_header = united.iloc[0]
united = united[1:]
united.columns = new_header
united.head()
```

[119]:

0	Date	Open	High	Low	Close	Adj Close	Volume
1	2016-05-02	46.610001	46.810001	45.810001	46.660000	46.660000	6626200
2	2016-05-03	47.000000	47.540001	46.150002	47.130001	47.130001	6726600
3	2016-05-04	46.709999	46.759998	45.150002	45.549999	45.549999	7184800
4	2016-05-05	45.599998	46.070000	44.959999	44.990002	44.990002	6037400
5	2016-05-06	44.799999	45.730000	44.009998	45.700001	45.700001	9247300

[120]:
```
united["Volume"] = united["Volume"].astype(str).astype(int)
united["Open"] = united["Open"].astype(str).astype(float)
united["High"] = united["High"].astype(str).astype(float)
united["Low"] = united["Low"].astype(str).astype(float)
united["Close"] = united["Close"].astype(str).astype(float)
united.dtypes
```

[120]:
```
0
Date          object
Open         float64
```

```
High          float64
Low           float64
Close         float64
Adj Close      object
Volume          int32
dtype: object
```

[121]:
```
united = united[~united.Date.str.contains('2016')]
united = united[~united.Date.str.contains('2017')]
united = united[~united.Date.str.contains('2018')]
united = united[~united.Date.str.contains('2019')]
united['Month'] = pd.DatetimeIndex(united['Date']).month
united['Day'] = pd.DatetimeIndex(united['Date']).day
united.head()
```

[121]:

0	Date	Open	High	Low	Close	Adj Close	\
925	2020-01-02	89.570000	90.570000	89.110001	89.739998	89.739998	
926	2020-01-03	86.800003	88.160004	86.260002	87.900002	87.900002	
927	2020-01-06	86.720001	88.070000	86.650002	87.699997	87.699997	
928	2020-01-07	87.410004	88.160004	86.739998	86.769997	86.769997	
929	2020-01-08	86.900002	88.449997	86.300003	87.300003	87.300003	

0	Volume	Month	Day
925	2769800	1	2
926	3562900	1	3
927	2652700	1	6
928	2581300	1	7
929	4152500	1	8

Amazon Stock - as a comparison

[122]:
```
amazon = pd.read_csv('data/airlines_travel/amazon.csv', sep= ',', thousands=',',␣
 ↪header= None)

# a summary of the travel restrictions file
amazon.head()
```

[122]:

0	1	2	3	4	5	6	\
0	NaN	Date	Open	High	Low	Close	Adj. Close	
1	0.0	NaN	NaN	NaN	NaN	NaN	NaN	
2	1.0	Jul 14, 2020	3,089.00	3,125.50	2,950.00	3,031.57	3,031.57	
3	2.0	Jul 13, 2020	3,251.06	3,344.29	3,068.39	3,104.00	3,104.00	
4	3.0	Jul 10, 2020	3,191.76	3,215.00	3,135.70	3,200.00	3,200.00	

0	7
0	Volume
1	NaN

```
2  3,745,145
3  7,689,300
4  5,486,000
```

[123]:
```
new_header = amazon.iloc[0]
amazon = amazon[2:]
amazon.columns = new_header
amazon.head()
```

[123]:
```
0  NaN          Date      Open      High       Low     Close Adj. Close  \
2  1.0  Jul 14, 2020  3,089.00  3,125.50  2,950.00  3,031.57   3,031.57
3  2.0  Jul 13, 2020  3,251.06  3,344.29  3,068.39  3,104.00   3,104.00
4  3.0  Jul 10, 2020  3,191.76  3,215.00  3,135.70  3,200.00   3,200.00
5  4.0  Jul 09, 2020  3,115.99  3,193.88  3,074.00  3,182.63   3,182.63
6  5.0  Jul 08, 2020  3,022.61  3,083.97  3,012.43  3,081.11   3,081.11

0     Volume
2  3,745,145
3  7,689,300
4  5,486,000
5  6,388,700
6  5,037,600
```

[124]:
```
amazon["Volume"] = amazon["Volume"].str.replace(",","").astype(float)
amazon["Open"] = amazon["Open"].str.replace(",","").astype(float)
amazon["High"] = amazon["High"].str.replace(",","").astype(float)
amazon["Low"] = amazon["Low"].str.replace(",","").astype(float)
amazon["Close"] = amazon["Close"].str.replace(",","").astype(float)
amazon.dtypes
```

[124]:
```
0
NaN           float64
Date           object
Open          float64
High          float64
Low           float64
Close         float64
Adj. Close     object
Volume        float64
dtype: object
```

[125]:
```
amazon = amazon[amazon.Date.str.contains('2020')]
amazon['Month'] = pd.DatetimeIndex(amazon['Date']).month
amazon['Day'] = pd.DatetimeIndex(amazon['Date']).day
amazon.head()
```

[125]:
```
0  NaN          Date     Open     High      Low    Close Adj. Close  \
2  1.0  Jul 14, 2020  3089.00  3125.50  2950.00  3031.57   3,031.57
3  2.0  Jul 13, 2020  3251.06  3344.29  3068.39  3104.00   3,104.00
4  3.0  Jul 10, 2020  3191.76  3215.00  3135.70  3200.00   3,200.00
5  4.0  Jul 09, 2020  3115.99  3193.88  3074.00  3182.63   3,182.63
6  5.0  Jul 08, 2020  3022.61  3083.97  3012.43  3081.11   3,081.11

0     Volume  Month  Day
2  3745145.0      7   14
3  7689300.0      7   13
4  5486000.0      7   10
5  6388700.0      7    9
6  5037600.0      7    8
```

S&P 500 Stock data - as a comparison

[126]:
```
sp500 = pd.read_csv('data/airlines_travel/SP500.csv', sep= ',', header= None)

# a summary of the travel restrictions file
sp500.head()
```

[126]:
```
             0            1            2            3            4  \
0         Date         Open         High          Low        Close
1   2000-01-03  1469.250000  1478.000000  1438.359985  1455.219971
2   2000-01-04  1455.219971  1455.219971  1397.430054  1399.420044
3   2000-01-05  1399.420044  1413.270020  1377.680054  1402.109985
4   2000-01-06  1402.109985  1411.900024  1392.099976  1403.449951

             5           6
0    Adj Close      Volume
1  1455.219971   931800000
2  1399.420044  1009000000
3  1402.109985  1085500000
4  1403.449951  1092300000
```

[127]:
```
new_header = sp500.iloc[0]
sp500 = sp500[1:]
sp500.columns = new_header
sp500.head()
```

[127]:
```
0         Date         Open         High          Low        Close  \
1   2000-01-03  1469.250000  1478.000000  1438.359985  1455.219971
2   2000-01-04  1455.219971  1455.219971  1397.430054  1399.420044
3   2000-01-05  1399.420044  1413.270020  1377.680054  1402.109985
4   2000-01-06  1402.109985  1411.900024  1392.099976  1403.449951
5   2000-01-07  1403.449951  1441.469971  1400.729980  1441.469971
```

```
0    Adj Close      Volume
1  1455.219971   931800000
2  1399.420044  1009000000
3  1402.109985  1085500000
4  1403.449951  1092300000
5  1441.469971  1225200000
```

[128]:
```
sp500["Volume"] = sp500["Volume"].astype(str).astype(np.int64)
sp500["Open"] = sp500["Open"].astype(str).astype(float)
sp500["High"] = sp500["High"].astype(str).astype(float)
sp500["Low"] = sp500["Low"].astype(str).astype(float)
sp500["Close"] = sp500["Close"].astype(str).astype(float)
sp500.dtypes
```

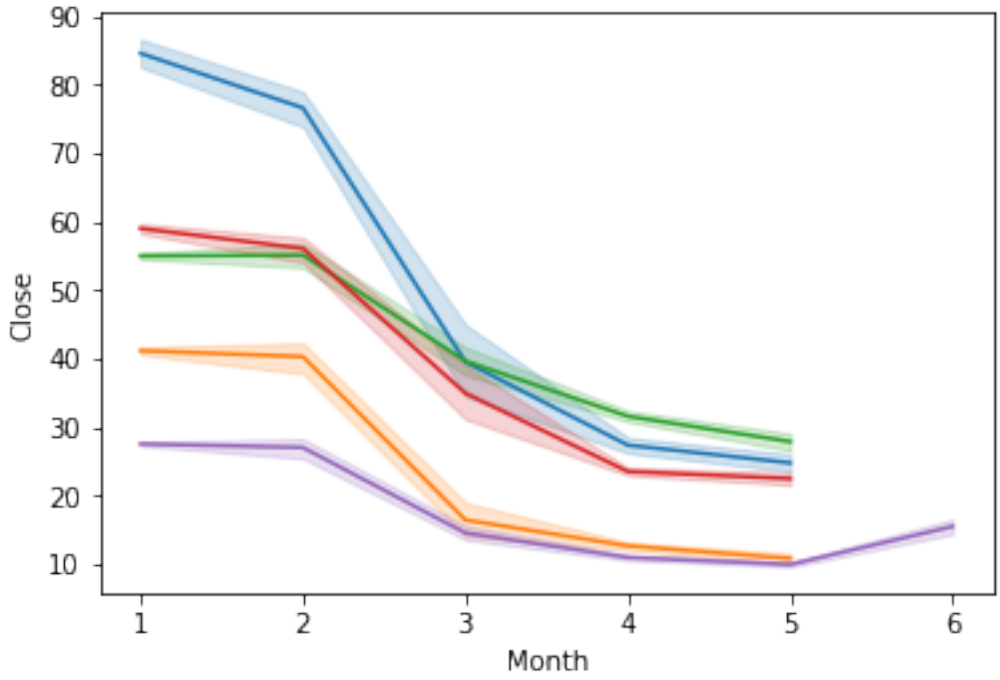
[128]:
```
0
Date          object
Open         float64
High         float64
Low          float64
Close        float64
Adj Close     object
Volume         int64
dtype: object
```

[129]:
```
sp500 = sp500[sp500.Date.str.contains('2020')]
sp500['Month'] = pd.DatetimeIndex(sp500['Date']).month
sp500['Day'] = pd.DatetimeIndex(sp500['Date']).day
sp500.head()
```

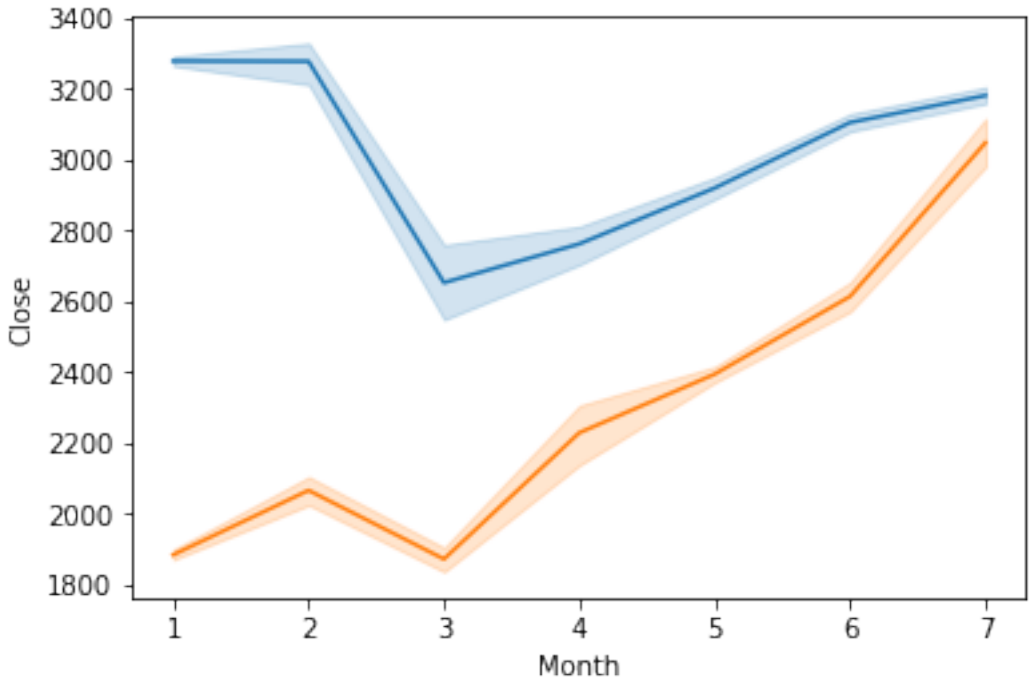
[129]:
```
0           Date         Open         High          Low        Close  \
5032  2020-01-02  3244.669922  3258.139893  3235.530029  3257.850098
5033  2020-01-03  3226.360107  3246.149902  3222.340088  3234.850098
5034  2020-01-06  3217.550049  3246.840088  3214.639893  3246.280029
5035  2020-01-07  3241.860107  3244.909912  3232.429932  3237.179932
5036  2020-01-08  3238.590088  3267.070068  3236.669922  3253.050049

0       Adj Close      Volume  Month  Day
5032  3257.850098  3458250000      1    2
5033  3234.850098  3461290000      1    3
5034  3246.280029  3674070000      1    6
5035  3237.179932  3420380000      1    7
5036  3253.050049  3720890000      1    8
```

Stock Market Graphs

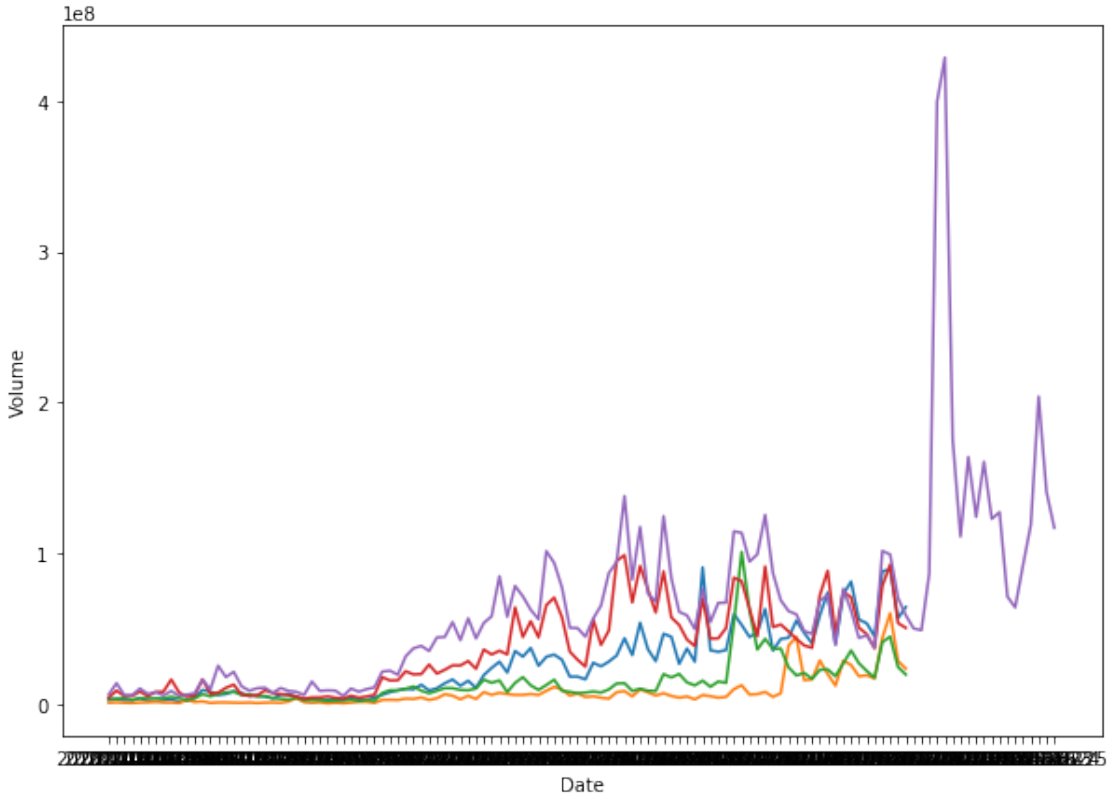
[130]:
```
# five airlines
ax = sns.lineplot(x="Month", y="Close", data=united)
ax = sns.lineplot(x="Month", y="Close", data=spirit_air)
ax = sns.lineplot(x="Month", y="Close", data=southwest)
ax = sns.lineplot(x="Month", y="Close", data=delta)
ax = sns.lineplot(x="Month", y="Close", data=american_air)
```



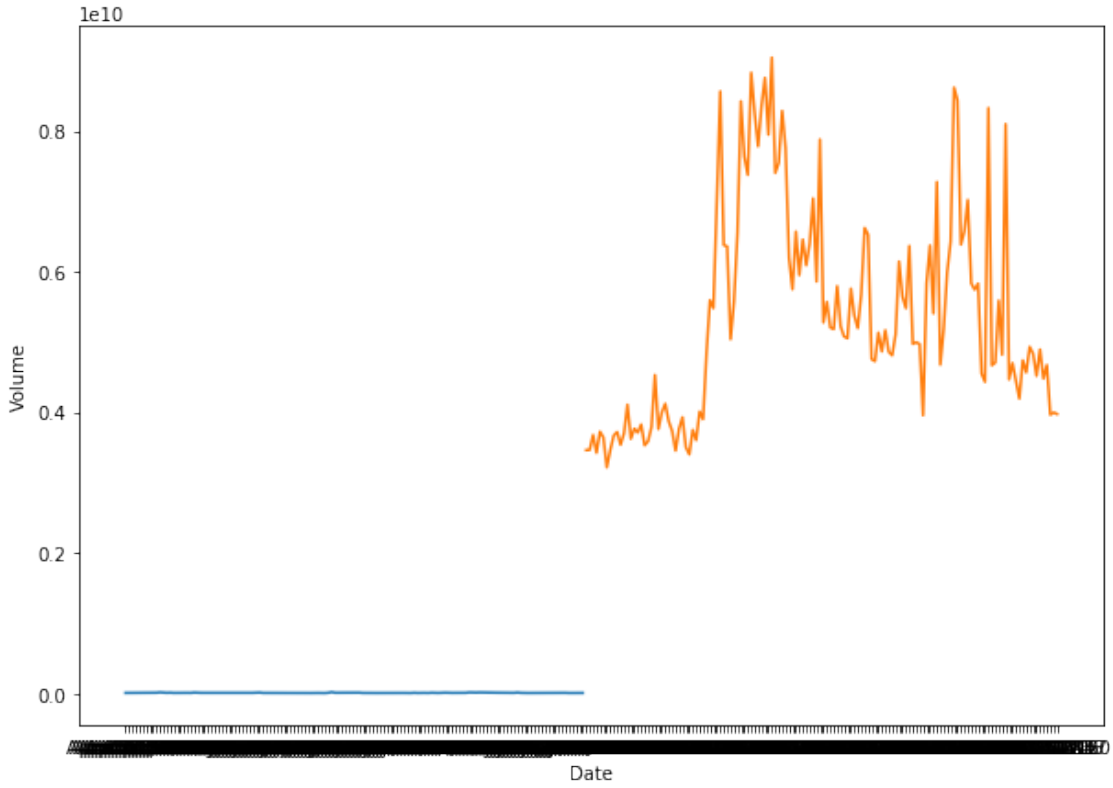
[131]:
```
# comparison
ax = sns.lineplot(x="Month", y="Close", data=sp500)
ax = sns.lineplot(x="Month", y="Close", data=amazon)
```



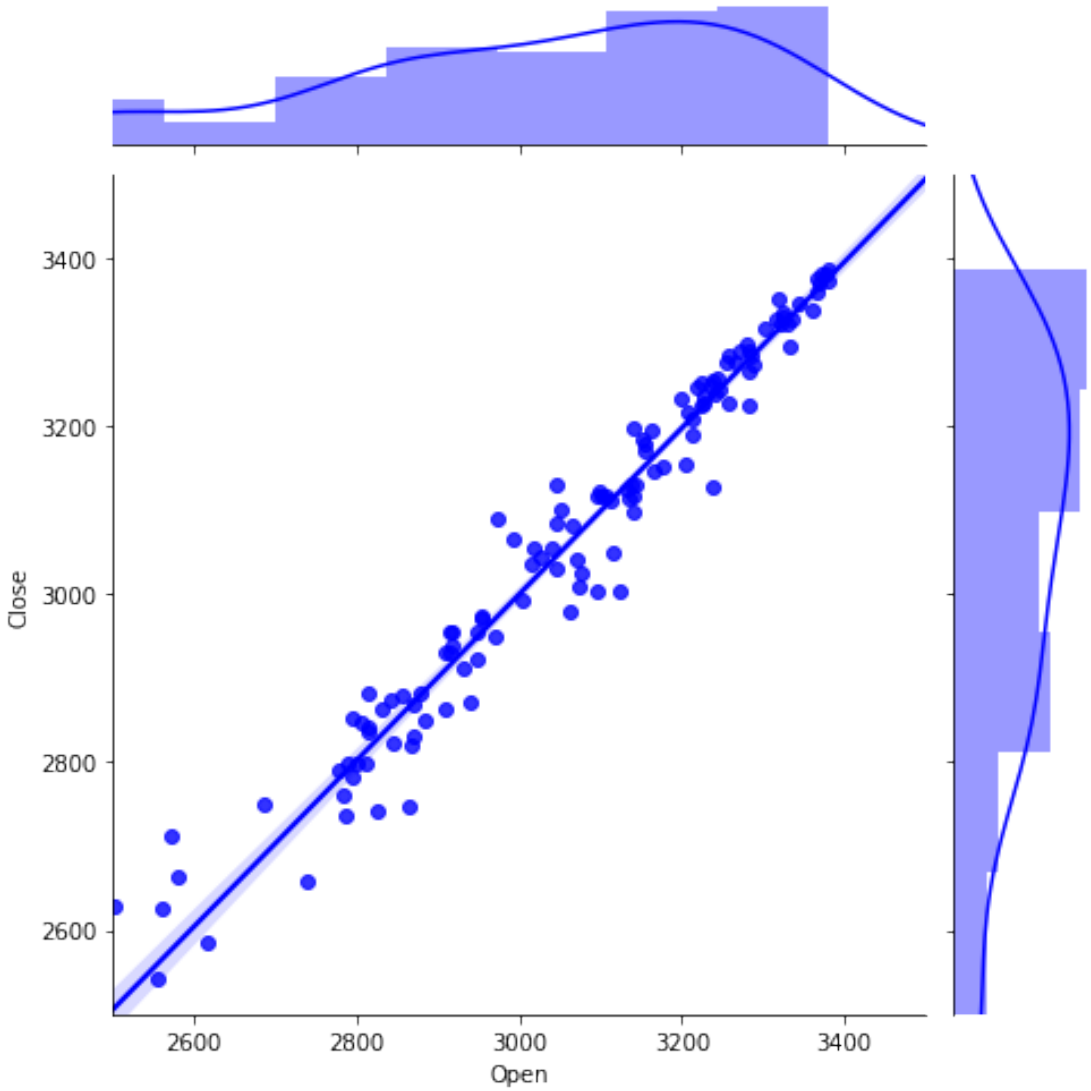
[132]:
```
fig, ax = plt.subplots(figsize=fig_dims)
ax = sns.lineplot(x="Date", y="Volume", data=united)
ax = sns.lineplot(x="Date", y="Volume", data=spirit_air)
ax = sns.lineplot(x="Date", y="Volume", data=southwest)
ax = sns.lineplot(x="Date", y="Volume", data=delta)
ax = sns.lineplot(x="Date", y="Volume", data=american_air)
```

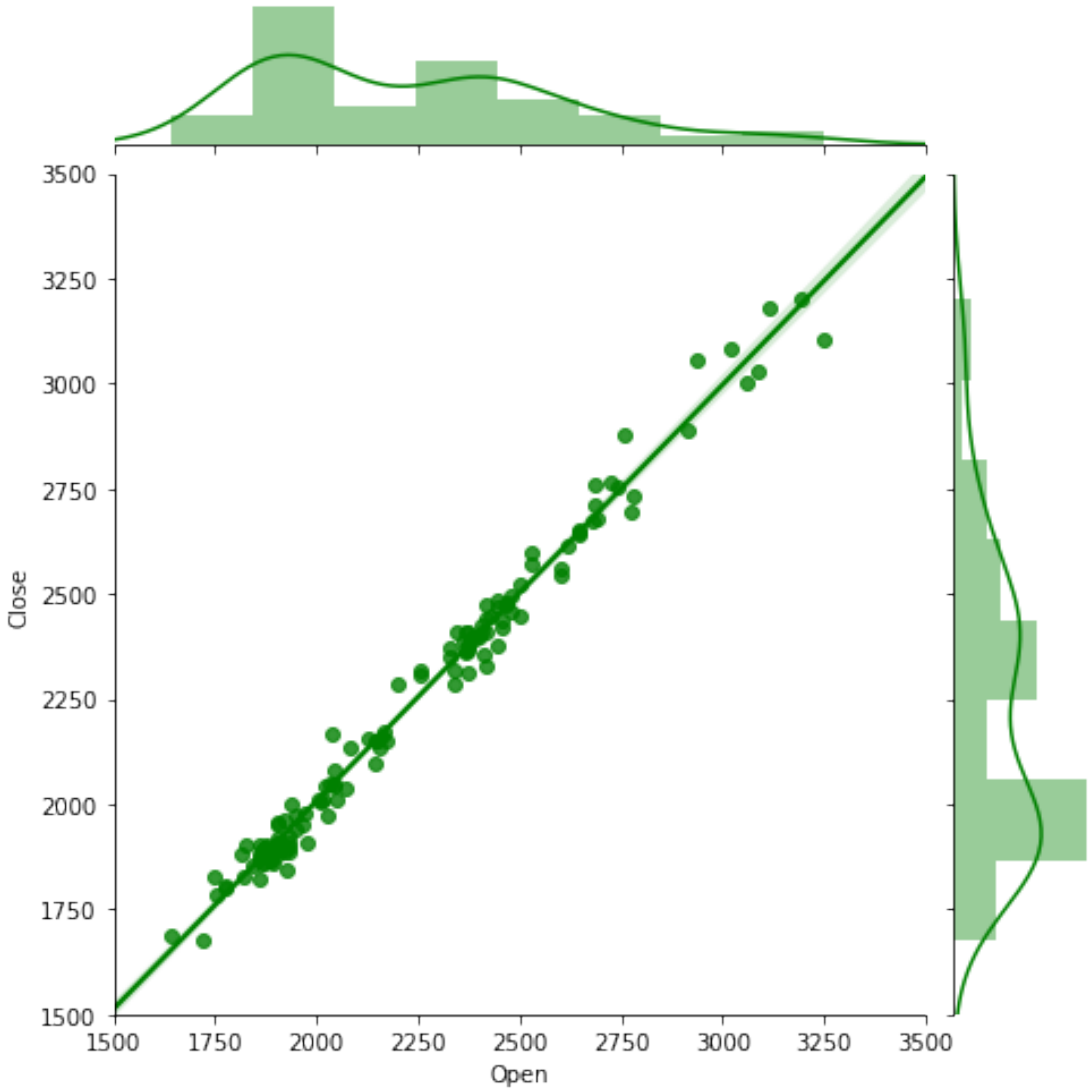


[133]:
```
fig, ax = plt.subplots(figsize=fig_dims)
ax = sns.lineplot(x="Date", y="Volume", data=amazon)
ax = sns.lineplot(x="Date", y="Volume", data=sp500)
```



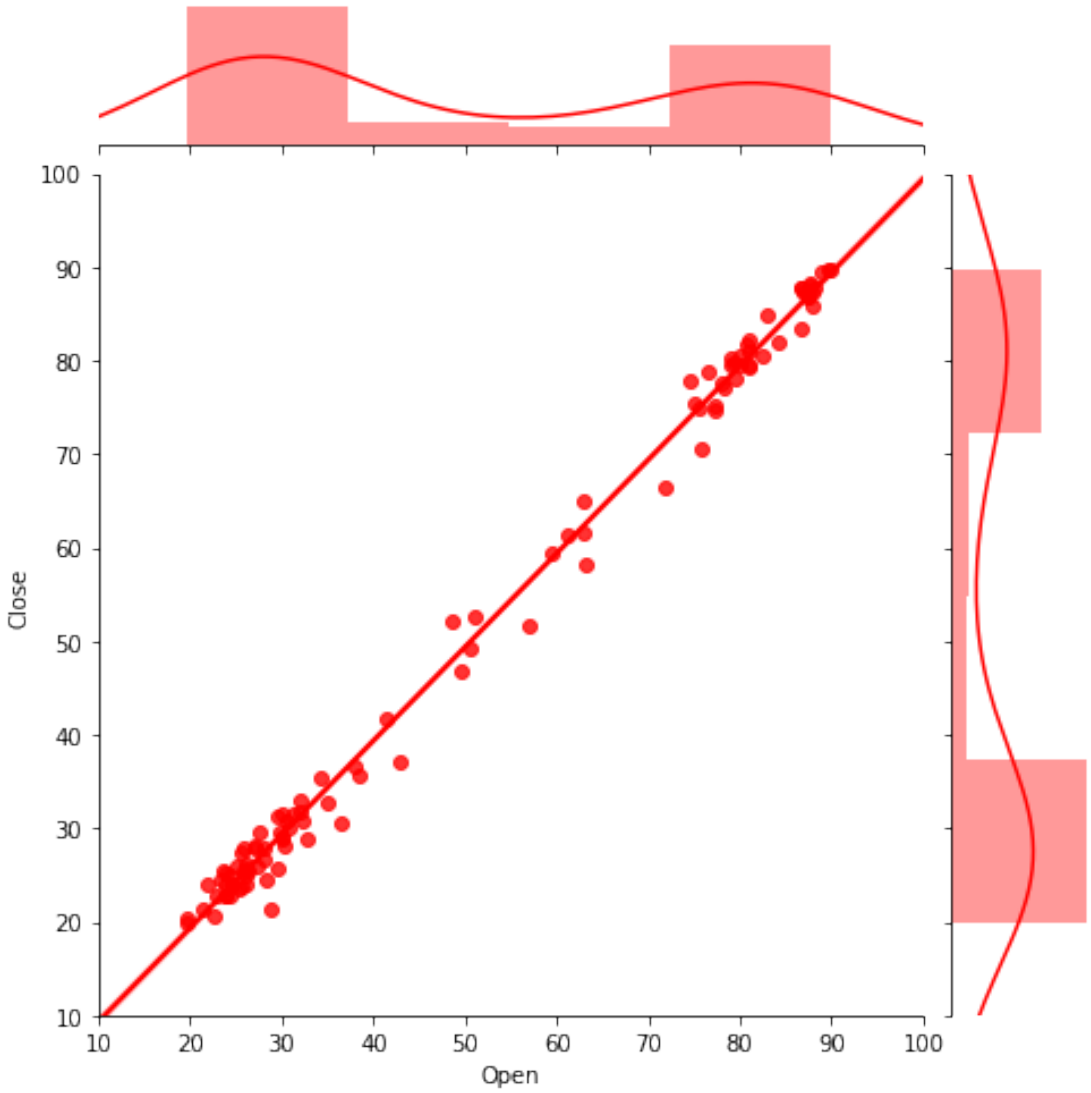
[134]:
```
ax = sns.jointplot("Open", "Close", data=sp500,
                   kind="reg", truncate=False,
                   xlim=(2500, 3500), ylim=(2500, 3500),
                   color="b", height=7)
ax = sns.jointplot("Open", "Close", data=amazon,
                   kind="reg", truncate=False,
                   xlim=(1500, 3500), ylim=(1500, 3500),
                   color="g", height=7)
```

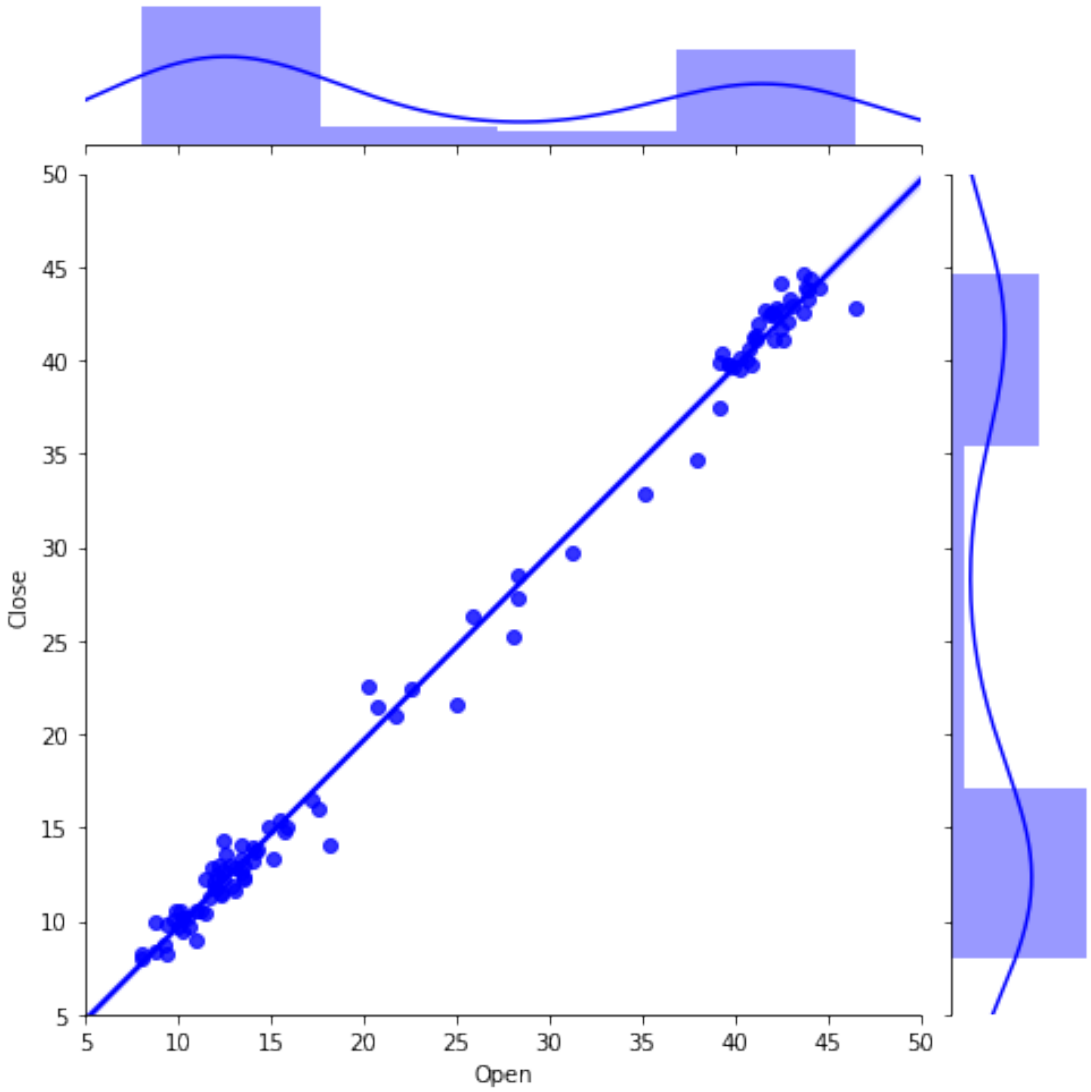

3400
3200
3000
2800
2600
Close
2600
2800
3000
3200
3400
Open

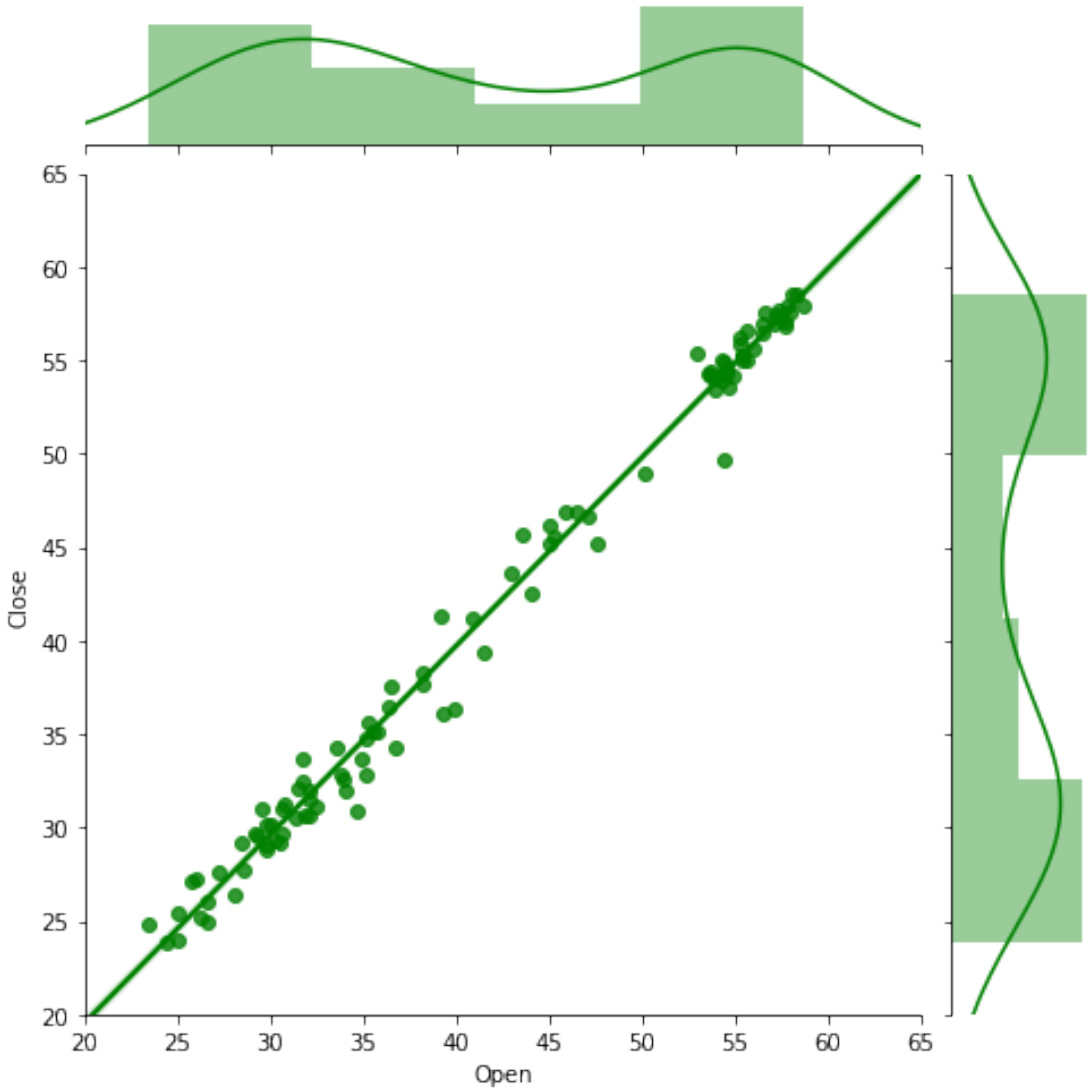


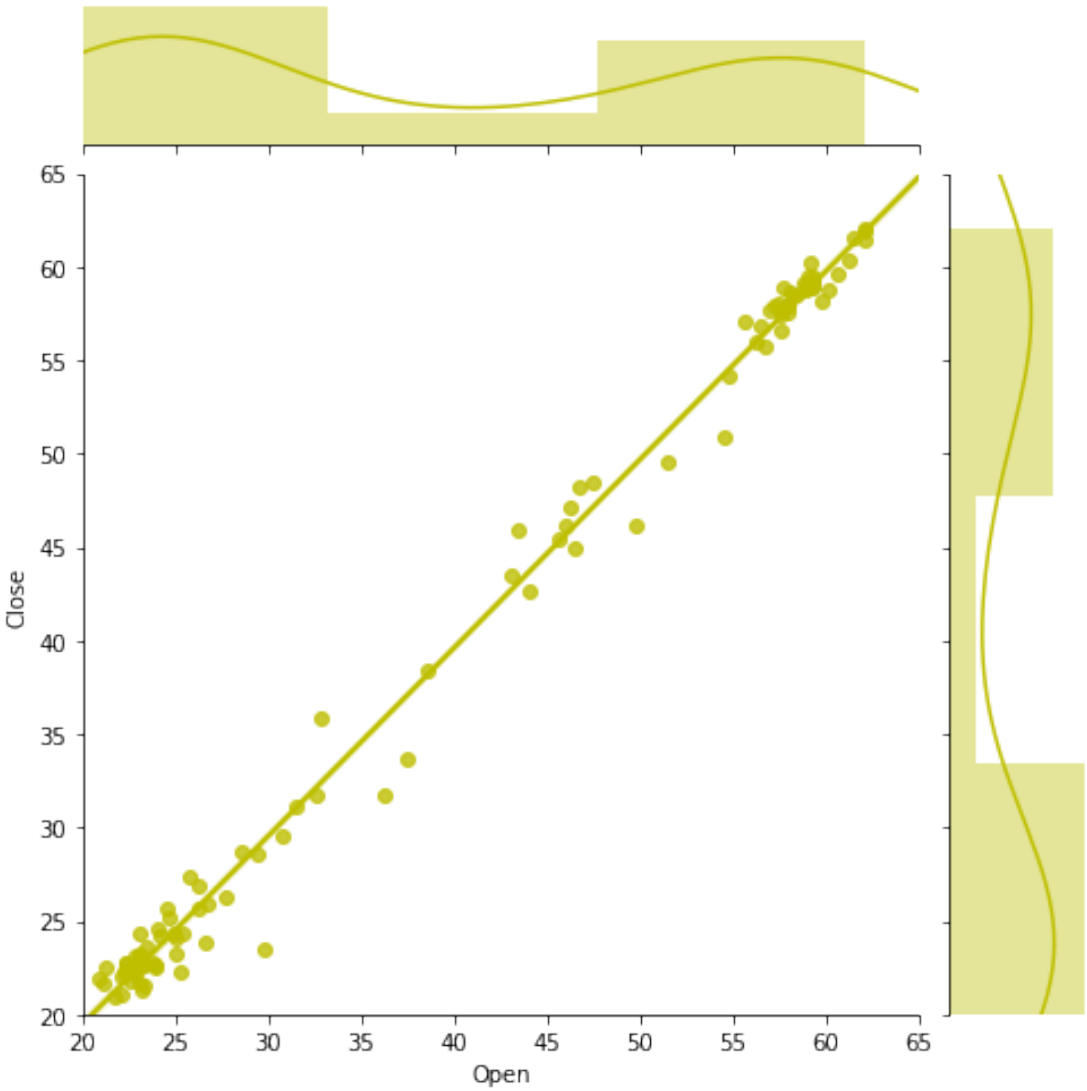
[135]:
```
ax = sns.jointplot("Open", "Close", data=united,
                   kind="reg", truncate=False,
                   xlim=(10, 100), ylim=(10, 100),
                   color="r", height=7)
ax = sns.jointplot("Open", "Close", data=spirit_air,
                   kind="reg", truncate=False,
                   xlim=(5, 50), ylim=(5, 50),
                   color="b", height=7)
ax = sns.jointplot("Open", "Close", data=southwest,
                   kind="reg", truncate=False,
                   xlim=(20, 65), ylim=(20, 65),
                   color="g", height=7)
ax = sns.jointplot("Open", "Close", data=delta,
```

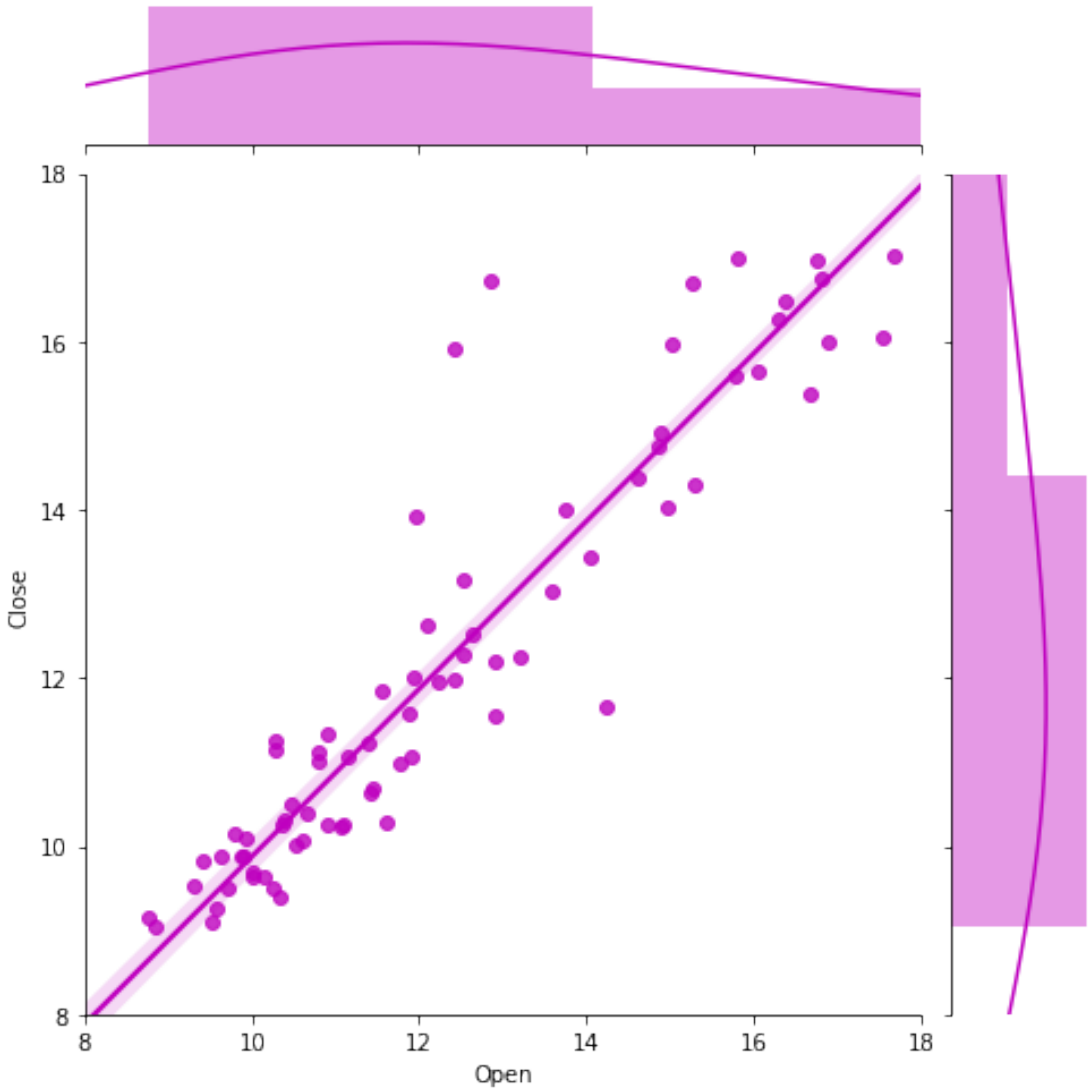
```
                    kind="reg", truncate=False,
                    xlim=(20, 65), ylim=(20, 65),
                    color="y", height=7)
ax = sns.jointplot("Open", "Close", data=american_air,
                    kind="reg", truncate=False,
                    xlim=(8, 18), ylim=(8, 18),
                    color="m", height=7)
```



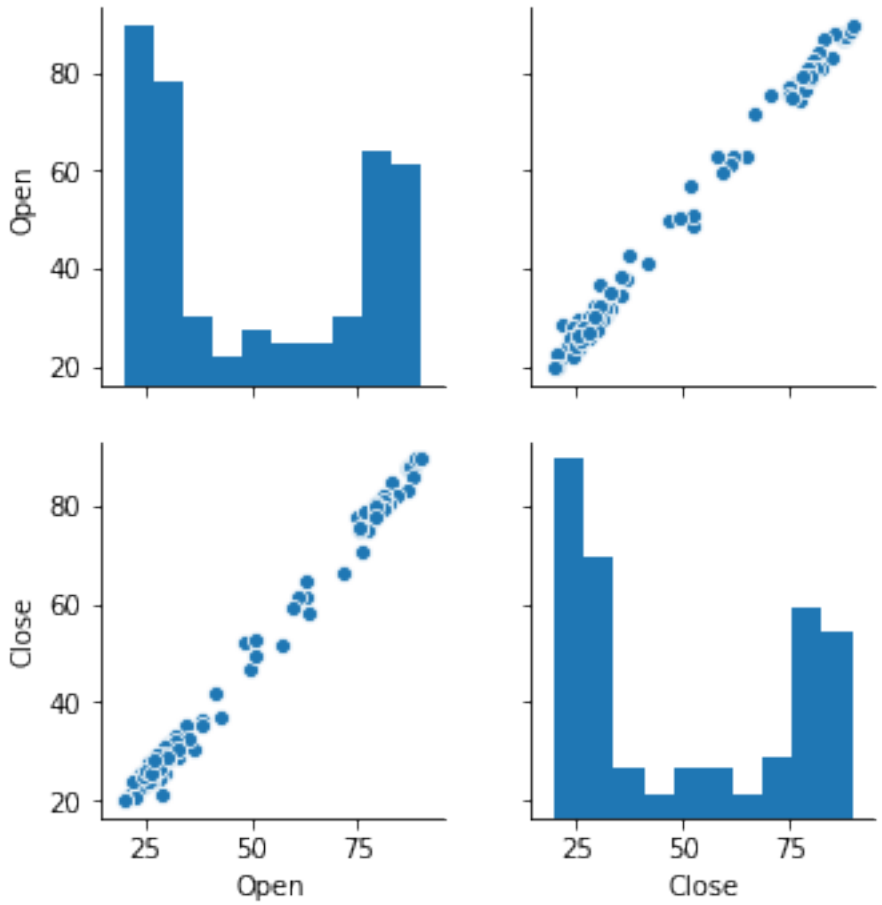

Close
Open
5
10
15
20
25
30
35
40
45
50

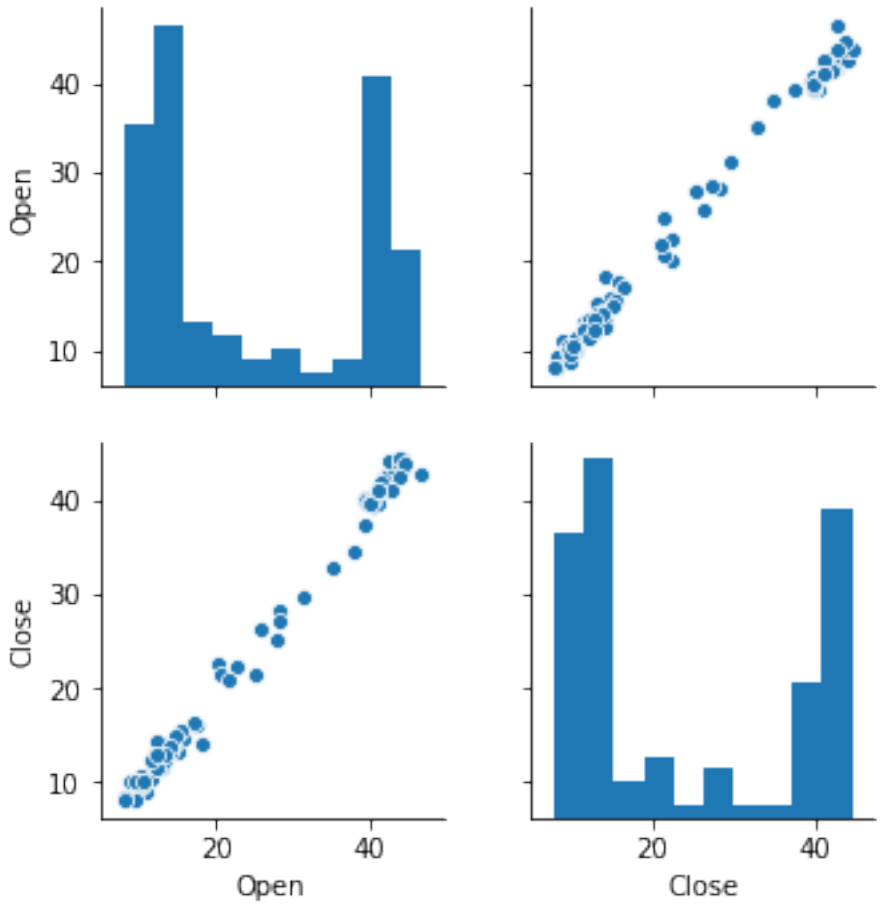

Close
Open
20
25
30
35
40
45
50
55
60
65

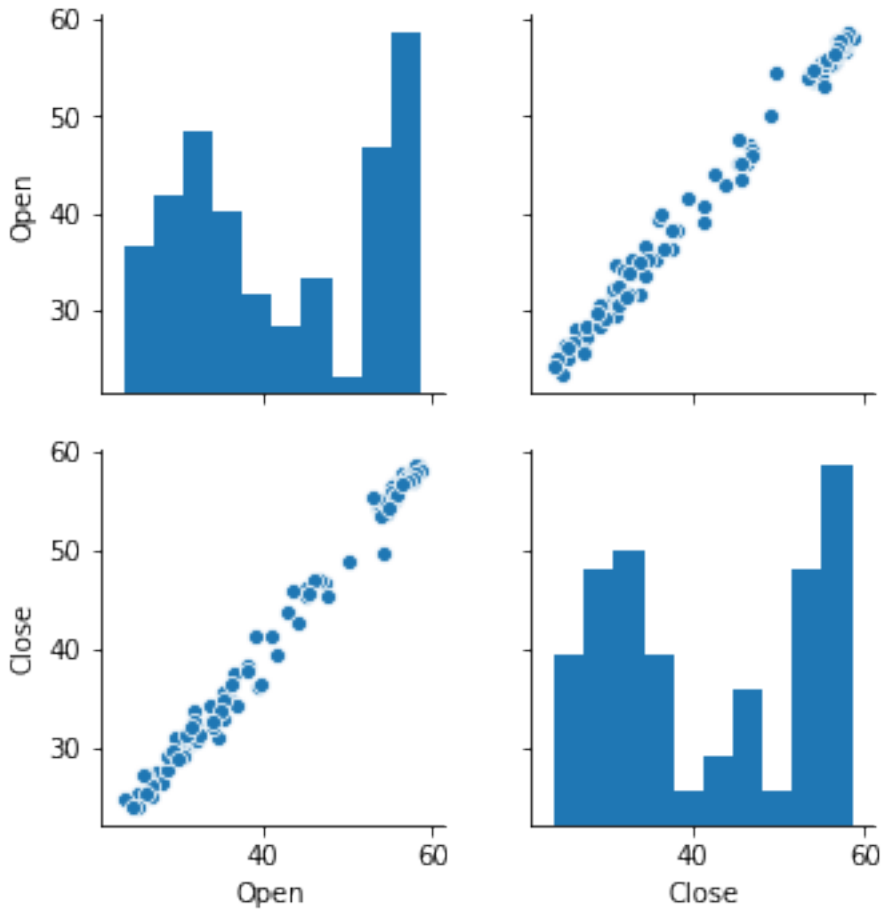

65
60
55
50
45
40
35
30
25
20
Close
20
25
30
35
40
45
50
55
60
65
Open

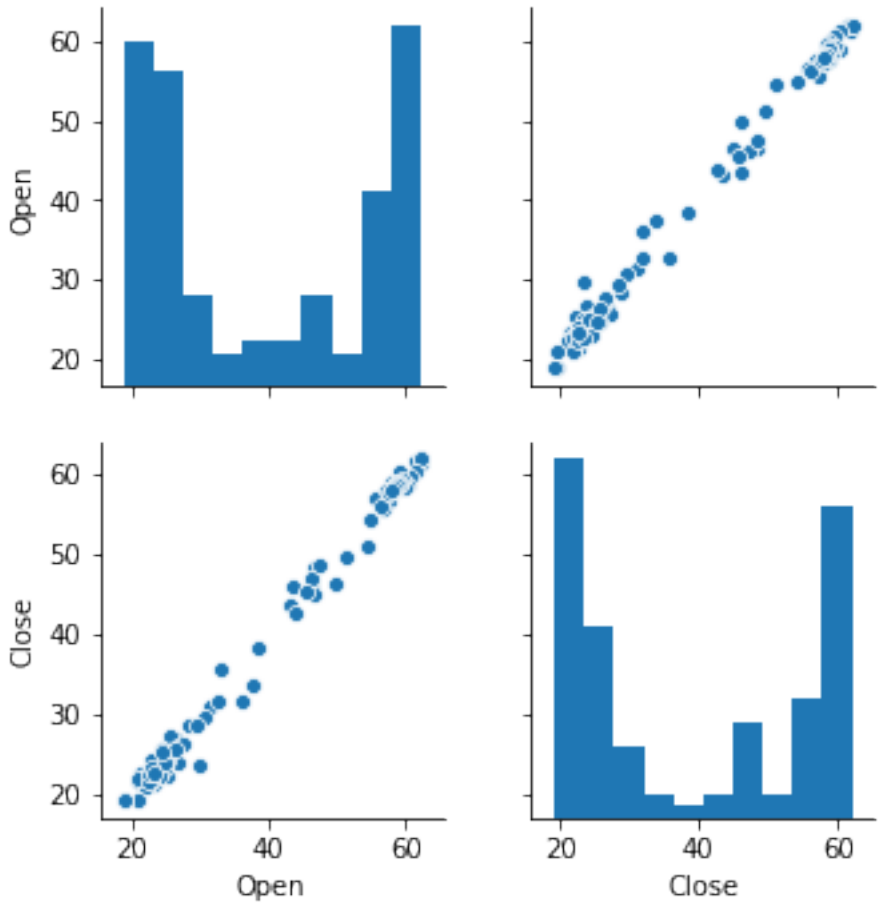


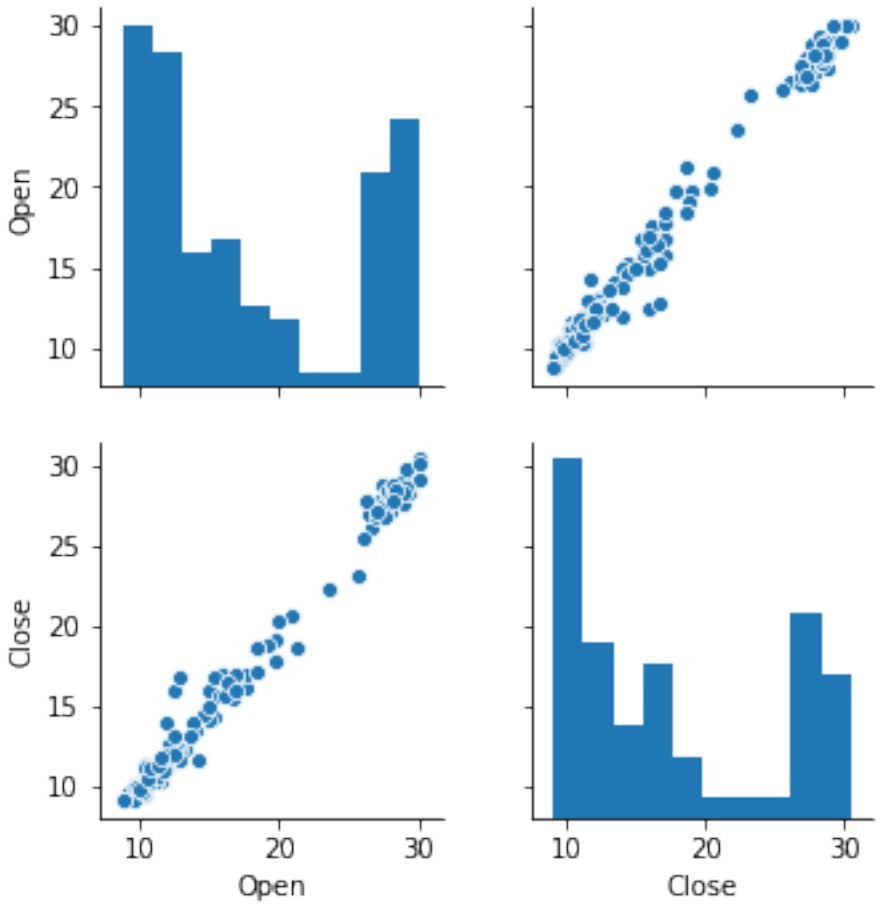
[136]:
```
ax = sns.pairplot(data=united, vars=["Open", "Close"])
ax = sns.pairplot(data=spirit_air, vars=["Open", "Close"])
ax = sns.pairplot(data=southwest, vars=["Open", "Close"])
ax = sns.pairplot(data=delta, vars=["Open", "Close"])
ax = sns.pairplot(data=american_air, vars=["Open", "Close"])
```


Open
Close
Open
Close
20
40
60
80
25
50
75

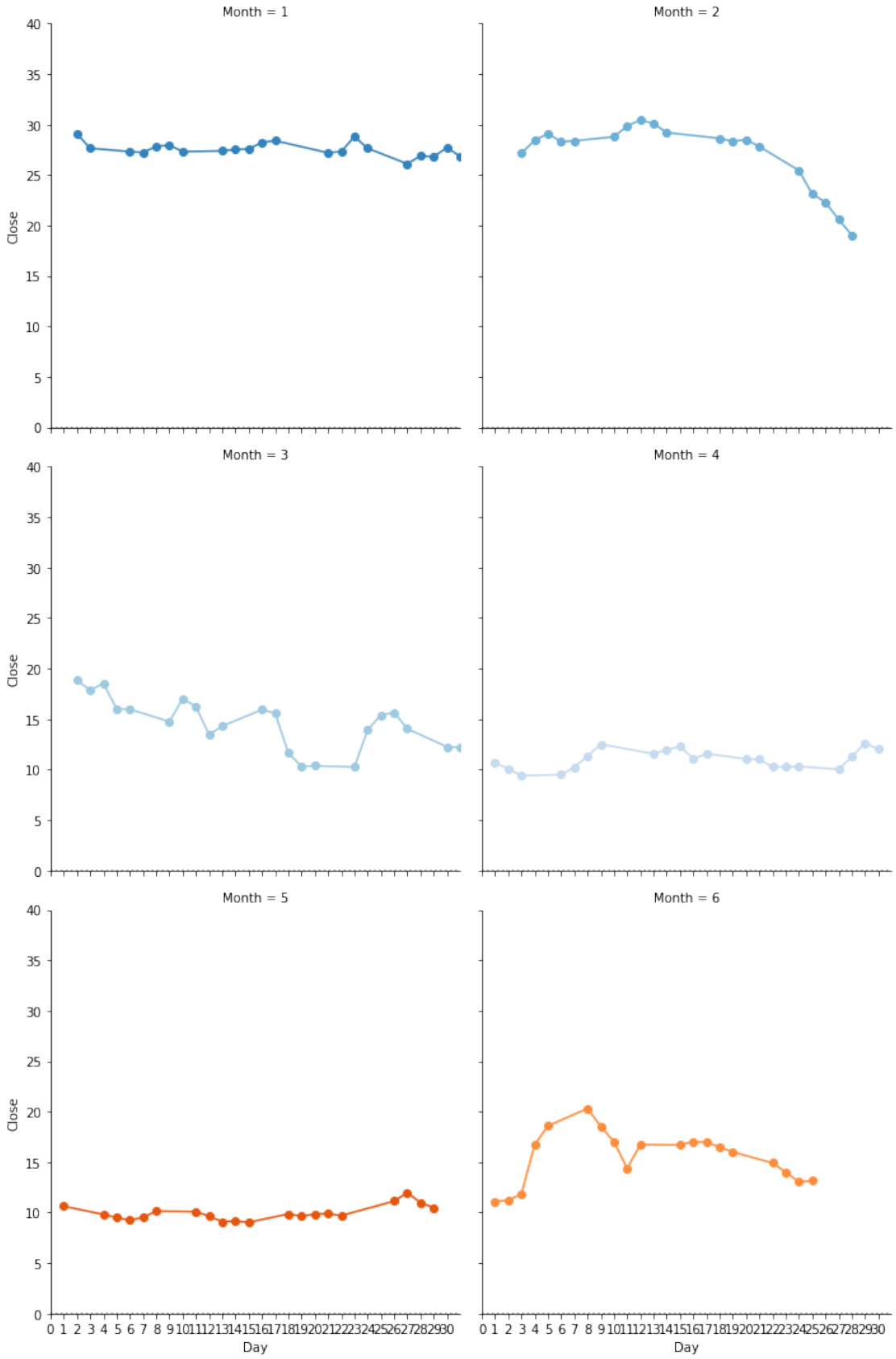

Open
Close
Open
Close
10
20
30
40
10
20
30
40
20
40
20
40


60
50
40
30
Open
60
50
40
30
Close
40
60
Open
40
60
Close


Open
Close
Open
Close
20
30
40
50
60



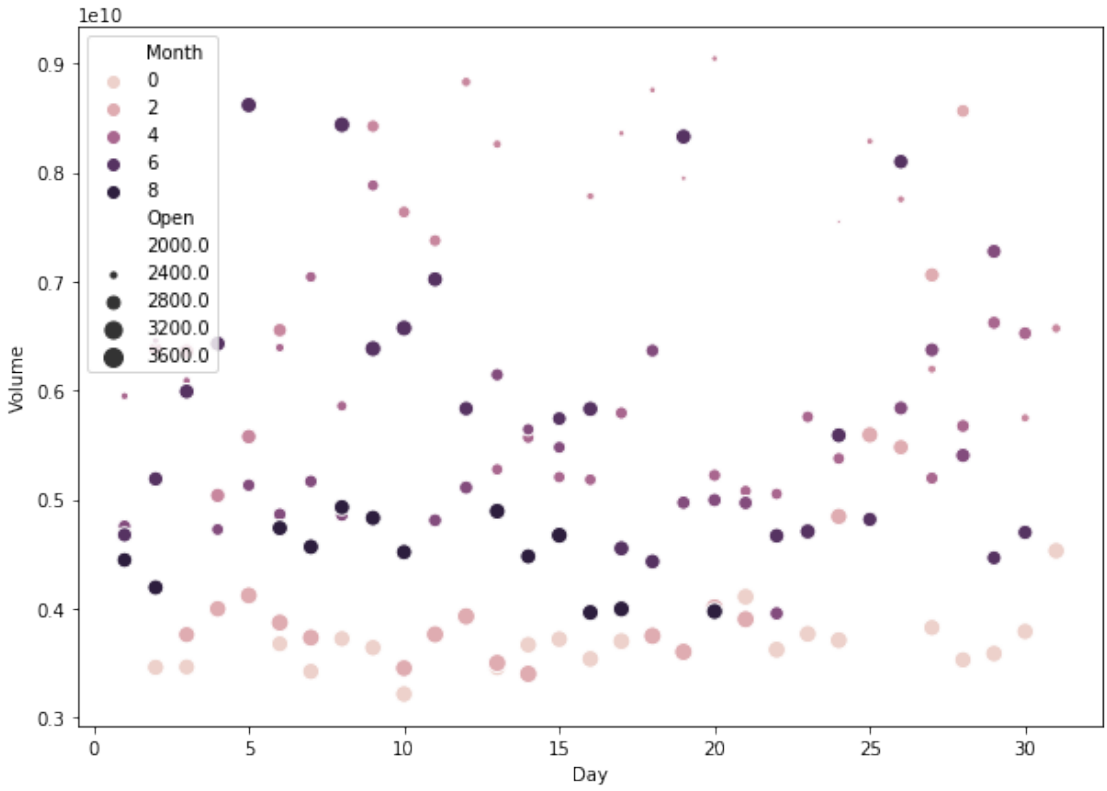
[137]:
```
# American air stocks by day and month
grid = sns.FacetGrid(american_air, col="Month", hue="Month", palette="tab20c",
                     col_wrap=2, height=5)
# Draw a horizontal line to show the starting point
grid.map(plt.axhline, y=0, ls=":", c=".5")

# Draw a line plot to show the trajectory of each random walk
grid.map(plt.plot, "Day", "Close", marker="o")

# Adjust the tick positions and labels
grid.set(xticks=np.arange(31), yticks=[0, 5, 10, 15, 20, 25, 30, 35, 40],
         xlim=(0, 31), ylim=(0, 40))

# Adjust the arrangement of the plots
grid.fig.tight_layout(w_pad=1)
```


Month = 1
Month = 2
Month = 3
Month = 4
Month = 5
Month = 6
Close
Day

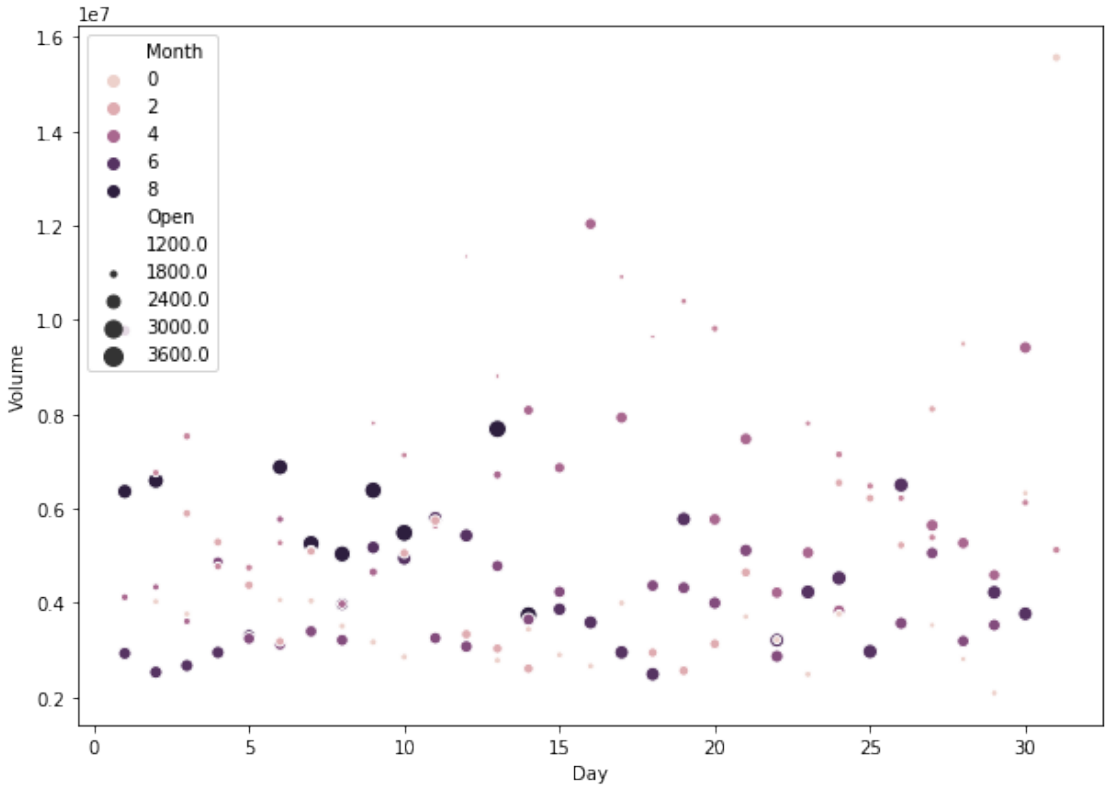
[138]:
```
# S&P 500 by day and month
grid = sns.FacetGrid(sp500, col="Month", hue="Month", palette="tab20c",
                     col_wrap=2, height=5)
# Draw a horizontal line to show the starting point
grid.map(plt.axhline, y=0, ls=":", c=".5")

# Draw a line plot to show the trajectory of each random walk
grid.map(plt.plot, "Day", "Close", marker="o")

# Adjust the tick positions and labels
grid.set(xticks=np.arange(31), yticks=[0, 500, 1000, 1500, 2000, 2500, 3000,␣
 ↪3500, 4000],
         xlim=(0, 31), ylim=(0, 4000))

# Adjust the arrangement of the plots
grid.fig.tight_layout(w_pad=1)
```


Month = 1
Month = 2
Month = 3
Month = 4
Month = 5
Month = 6
Month = 7
Close
Day
0
500
1000
1500
2000
2500
3000
3500
4000
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

[139]:
```
fig, ax = plt.subplots(figsize=fig_dims)
ax = sns.scatterplot(x="Day", y="Volume", hue="Month", size="Open",
                     sizes=(0,100), linewidth=1, data=sp500)
```



[140]:
```
fig, ax = plt.subplots(figsize=fig_dims)
ax = sns.scatterplot(x="Day", y="Volume", hue="Month", size="Open",
                     sizes=(0,100), linewidth=1, data=amazon)
```



Stat Analysis

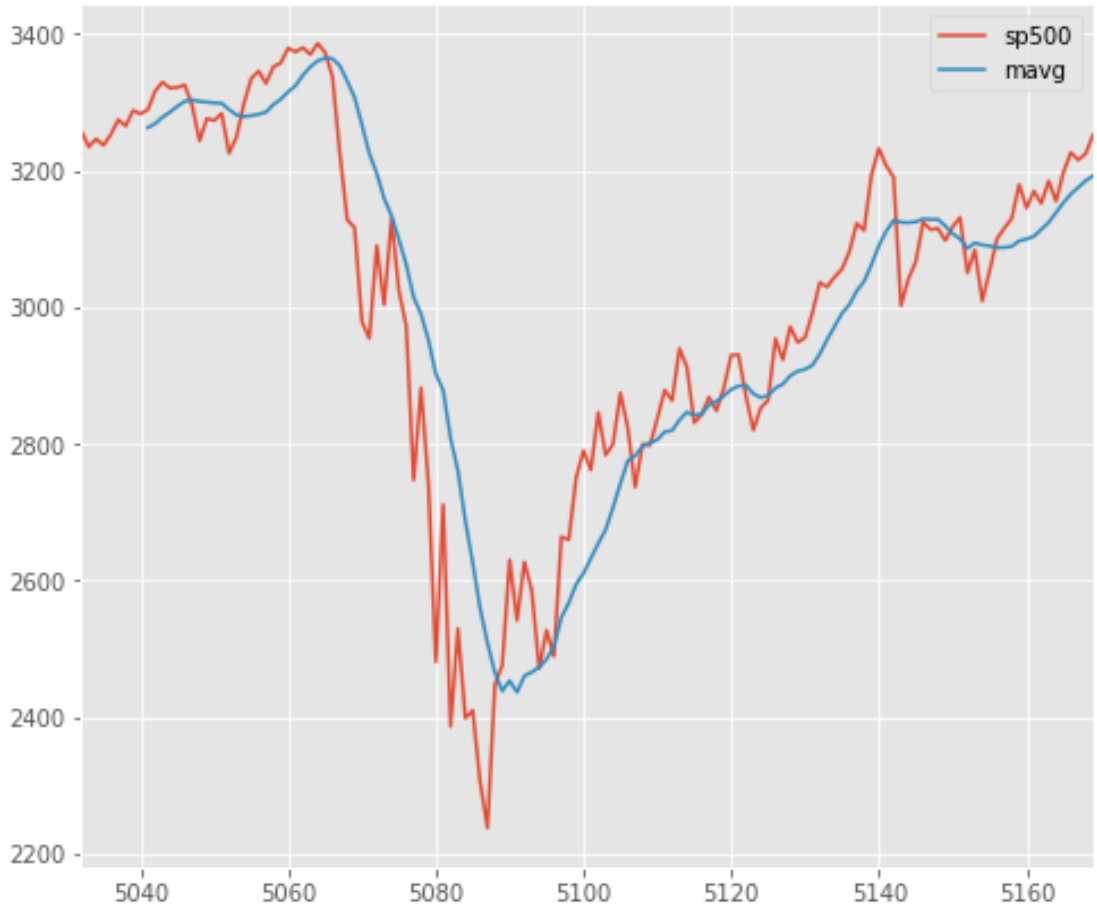
[141]:
```
close_px = sp500['Close']
mavg = close_px.rolling(window=10).mean()
print(mavg)
```

```
5032            NaN
5033            NaN
5034            NaN
5035            NaN
5036            NaN
5037            NaN
5038            NaN
5039            NaN
5040            NaN
5041    3262.983008
5042    3268.879004
5043    3278.356006
5044    3285.807007
5045    3294.264014
5046    3301.513013
5047    3303.590015
```

```
5048    3301.417993
5049    3300.229004
5050    3299.254004
5051    3298.690991
5052    3289.561987
5053    3281.491968
5054    3279.171973
5055    3280.465967
5056    3282.489966
5057    3285.713965
5058    3296.559985
5059    3304.710986
5060    3315.315991
5061    3324.343994
           ...
5140    3090.003003
5141    3111.543994
5142    3126.944995
5143    3124.182007
5144    3123.882007
5145    3124.968018
5146    3129.360010
5147    3128.421997
5148    3128.720996
5149    3119.102002
5150    3107.649024
5151    3100.060034
5152    3086.079053
5153    3094.245044
5154    3091.019043
5155    3089.684033
5156    3087.239038
5157    3087.476050
5158    3088.943042
5159    3097.141040
5160    3099.887036
5161    3103.752026
5162    3113.924023
5163    3124.052026
5164    3138.669019
5165    3153.097022
5166    3165.724024
5167    3175.695020
5168    3185.167017
5169    3192.379028
Name: Close, Length: 138, dtype: float64
```

[142]:
```
%matplotlib inline

# Adjusting the size of matplotlib
mpl.rc('figure', figsize=(8, 7))
mpl.__version__

# Adjusting the style of matplotlib
style.use('ggplot')

close_px.plot(label='sp500')
mavg.plot(label='mavg')
plt.legend()
```

[142]:
```
<matplotlib.legend.Legend at 0x1a9aae38358>
```



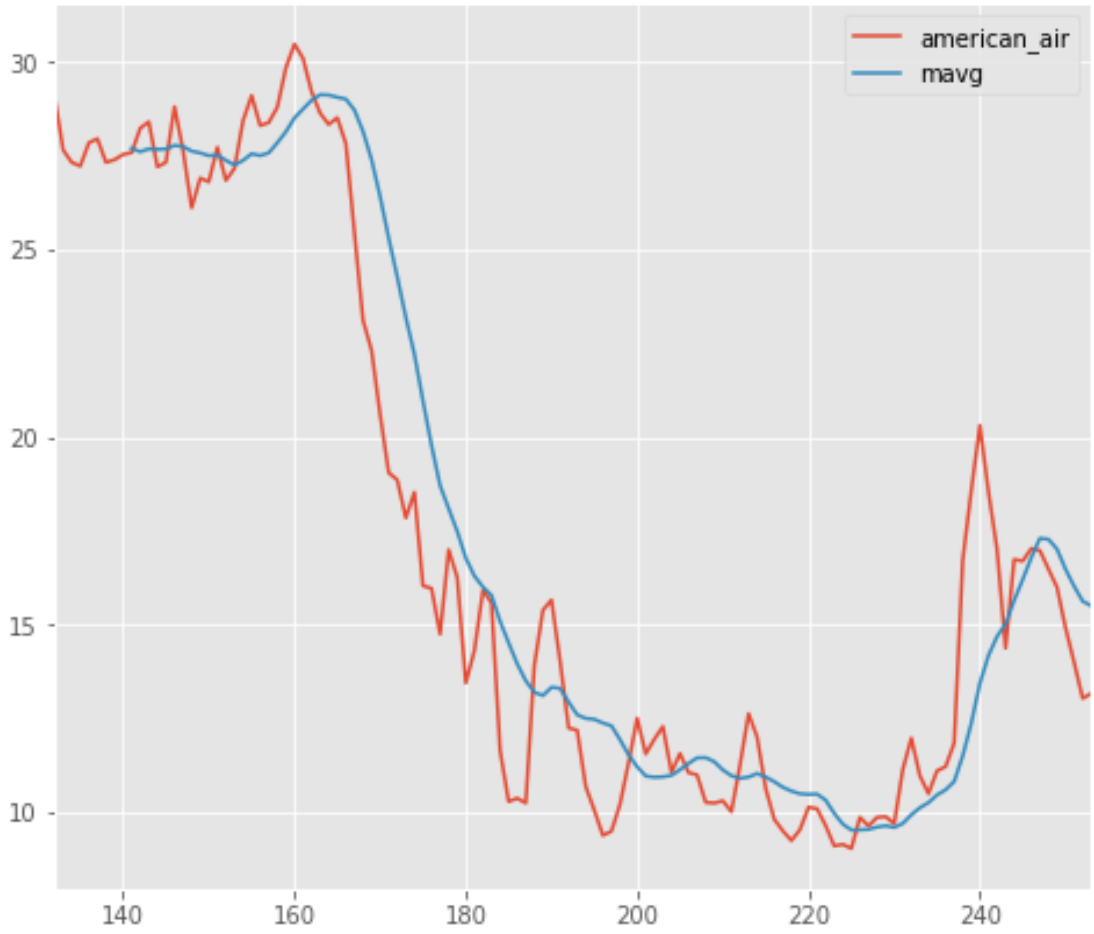
[143]:
```
close_px = american_air['Close']
mavg = close_px.rolling(window=10).mean()
print(mavg)
```

```
132       NaN
133       NaN
134       NaN
135       NaN
136       NaN
137       NaN
138       NaN
139       NaN
140       NaN
141    27.689
142    27.603
143    27.678
144    27.666
145    27.676
146    27.772
147    27.741
148    27.620
149    27.571
150    27.498
151    27.512
152    27.373
153    27.249
154    27.372
155    27.550
156    27.500
157    27.574
158    27.842
159    28.136
160    28.503
161    28.740
        ...
224     9.691
225     9.531
226     9.536
227     9.549
228     9.611
229     9.646
230     9.602
231     9.706
232     9.939
233    10.126
234    10.261
235    10.468
236    10.603
237    10.824
238    11.509
239    12.379
240    13.440
```

```
241    14.181
242    14.685
243    15.025
244    15.649
245    16.208
246    16.789
247    17.302
248    17.279
249    17.020
250    16.481
251    16.026
252    15.628
253    15.507
Name: Close, Length: 122, dtype: float64
```

[144]:
```
%matplotlib inline

# Adjusting the size of matplotlib
mpl.rc('figure', figsize=(8, 7))
mpl.__version__

# Adjusting the style of matplotlib
style.use('ggplot')

close_px.plot(label='american_air')
mavg.plot(label='mavg')
plt.legend()
```

[144]:
```
<matplotlib.legend.Legend at 0x1a9aaeeed68>
```



4 COVID Tests

[145]:
```
#Import WHO data for COVID Cases and deaths
filePathWHO = 'data/WHO-COVID-19-global-data.csv'
who = pd.read_csv(filePathWHO)
#includes new cases, new deaths, total cases, and total deaths for most countries
```

[146]:
```
#Import data from Our World in Data for COVID Tests
filePathTestsFull = 'data/full-list-total-tests-for-covid-19.csv'
covid_tests = pd.read_csv(filePathTestsFull)

#source: https://ourworldindata.org/grapher/full-list-total-tests-for-covid-19
```

[147]:
```
#Display the data format of the WHO dataset
who.head()
```

[147]:
```
  Date_reported  Country_code      Country  WHO_region   New_cases  \
0    2020-02-24            AF  Afghanistan        EMRO           1
1    2020-02-25            AF  Afghanistan        EMRO           0
2    2020-02-26            AF  Afghanistan        EMRO           0
3    2020-02-27            AF  Afghanistan        EMRO           0
4    2020-02-28            AF  Afghanistan        EMRO           0

    Cumulative_cases   New_deaths   Cumulative_deaths
0                  1            0                   0
1                  1            0                   0
2                  1            0                   0
3                  1            0                   0
4                  1            0                   0
```

[148]:
```
#Display the data format of the COVID Tests dataset
covid_tests.head()
```

[148]:
```
      Entity Code          Date  Total tests
0  Argentina  ARG   Apr 8, 2020        13330
1  Argentina  ARG   Apr 9, 2020        14850
2  Argentina  ARG  Apr 10, 2020        16379
3  Argentina  ARG  Apr 11, 2020        18027
4  Argentina  ARG  Apr 13, 2020        19758
```

[149]:
```
who['Date_reported'] =  pd.to_datetime(who['Date_reported'], format='%Y/%m/%d')
covid_tests['Date'] = pd.to_datetime(covid_tests.Date)
#convert the date format of covid_tests to be the same as in WHO data
```

[150]:
```
#Change the country naming scheme to be identical between the two datasets
who = who.replace(['United States of America',
                   'Bolivia (Plurinational State of)',
                   'Czechia','Iran (Islamic Republic of)',
                   'Russian Federation', 'Republic of Korea',
                   'The United Kingdom',
                   'Viet Nam'], ['United States',
                                 'Bolivia',
                                 'Czech Republic',
                                 'Iran',
                                 'Russia',
                                 'South Korea',
                                 'United Kingdom',
                                 'Vietnam'])
covid_tests = covid_tests.replace(['France, tests performed',
 'Ghana, people tested',
 'India, people tested',
 'Singapore, samples tested',
 'Thailand, people tested'], ['France',
```

```
                                'Ghana',
                                'India',
                                'Singapore',
                                'Thailand'])
```

[151]:
```
#Merge the WHO dataset and the COVID Tests dataset into one table, linked on the␣
 ↪name of the country and the date reported
who_tests_merged = covid_tests.merge(who, left_on=['Entity','Date'], right_on=['␣
 ↪Country','Date_reported'])
who_tests_merged = who_tests_merged.drop(['Entity', 'Date_reported', '␣
 ↪Country_code'], axis=1)
#Add a "new tests" category because only cumulative tests are in the dataset
who_tests_merged['New tests'] = 0
```

[152]:
```
#Using the total tests field to check for the number of new tests on each day in␣
 ↪the table
for i in range(who_tests_merged[' Country'].size):
    if i>0:
        if (who_tests_merged[' Country'][i-1] == who_tests_merged['␣
 ↪Country'][i]):
            who_tests_merged['New tests'][i] = (who_tests_merged['Total␣
 ↪tests'][i]-who_tests_merged['Total tests'][i-1])
#Note, takes ~3 min to run as is
```

```
C:\Users\nic_v\Anaconda3\lib\site-packages\ipykernel_launcher.py:5:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  """
```

[153]:
```
#Output Pearson and Spearman values for a variety of combinations from the␣
 ↪merged table generated
#Checking correlation of testing to both overall/new cases and deaths
xs = who_tests_merged[' Cumulative_cases']
ys = who_tests_merged['Total tests']
print('Pearson value for total cases and total tests of all countries:')
print(stats.pearsonr(xs,ys)[0])
print('Spearman value for total cases and total tests of all countries:')
print(stats.spearmanr(xs,ys)[0])
```

```
Pearson value for total cases and total tests of all countries:
0.9222211340936356
Spearman value for total cases and total tests of all countries:
0.8751822769092304
```

[154]:
```
xsr = who_tests_merged[' New_cases']
ysr = who_tests_merged['New tests']
print('Pearson value for new cases and new tests of all countries:')
print(stats.pearsonr(xsr,ysr)[0])
print('Spearman value for new cases and new tests of all countries:')
print(stats.spearmanr(xsr,ysr)[0])
```

```
Pearson value for new cases and new tests of all countries:
0.7691987665230835
Spearman value for new cases and new tests of all countries:
0.6607317347947645
```

[155]:
```
xus = who_tests_merged.loc[who_tests_merged[' Country'] == 'United States'][' ↪Cumulative_cases']
yus = who_tests_merged.loc[who_tests_merged[' Country'] == 'United ↪States']['Total tests']
print('Pearson value for total cases and total tests in the US:')
print(stats.pearsonr(xus,yus)[0])
print('Spearman value for total cases and total tests in the US:')
print(stats.spearmanr(xus,yus)[0])
```

```
Pearson value for total cases and total tests in the US:
0.9741862456385783
Spearman value for total cases and total tests in the US:
0.9999871245279883
```

[156]:
```
xus2 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United States'][' ↪New_cases']
yus2 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United ↪States']['New tests']
print('Pearson value for new cases and new tests in the US:')
print(stats.pearsonr(xus2,yus2)[0])
print('Spearman value for new cases and new tests in the US:')
print(stats.spearmanr(xus2,yus2)[0])
```

```
Pearson value for new cases and new tests in the US:
0.6854279491935551
Spearman value for new cases and new tests in the US:
0.5318661294857983
```

[157]:
```
xnz = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand'][' ↪Cumulative_cases']
ynz = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand']['Total ↪tests']
print('Pearson value for total cases and total tests in New Zealand:')
print(stats.pearsonr(xnz,ynz)[0])
print('Spearman value for total cases and total tests in New Zealand:')
```

```
print(stats.spearmanr(xnz,ynz)[0])
```

```
Pearson value for total cases and total tests in New Zealand:
0.7141574962311737
Spearman value for total cases and total tests in New Zealand:
0.99601154367086
```

[158]:
```
xnz2 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand'][' ␣
 ↪New_cases']
ynz2 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand']['New␣
 ↪tests']
print('Pearson value for new cases and new tests in New Zealand:')
print(stats.pearsonr(xnz2,ynz2)[0])
print('Spearman value for new cases and new tests in New Zealand:')
print(stats.spearmanr(xnz2,ynz2)[0])
```

```
Pearson value for new cases and new tests in New Zealand:
-0.1821895929177977
Spearman value for new cases and new tests in New Zealand:
0.0021873867473263258
```

[159]:
```
xs5 = who_tests_merged[' Cumulative_deaths']
ys5 = who_tests_merged['Total tests']
print('Pearson value for total deaths and total tests of all countries:')
print(stats.pearsonr(xs5,ys5)[0])
print('Spearman value for total deaths and total tests of all countries:')
print(stats.spearmanr(xs5,ys5)[0])
```

```
Pearson value for total deaths and total tests of all countries:
0.8133920066782202
Spearman value for total deaths and total tests of all countries:
0.8153897403814283
```

[160]:
```
xus3 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United States'][' ␣
 ↪Cumulative_deaths']
yus3 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United␣
 ↪States']['Total tests']
print('Pearson value for total deaths and total tests in the US:')
print(stats.pearsonr(xus3,yus3)[0])
print('Spearman value for total deaths and total tests in the US:')
print(stats.spearmanr(xus3,yus3)[0])
```

```
Pearson value for total deaths and total tests in the US:
0.9253135434226868
Spearman value for total deaths and total tests in the US:
0.9999656651869272
```

[161]:
```
xnz3 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand']['␣
 ↪Cumulative_deaths']
ynz3 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New␣
 ↪Zealand']['Total tests']
print('Pearson value for total deaths and total tests in NZ:')
print(stats.pearsonr(xus3,yus3)[0])
print('Spearman value for total deaths and total tests in NZ:')
print(stats.spearmanr(xus3,yus3)[0])
```

```
Pearson value for total deaths and total tests in NZ:
0.9253135434226868
Spearman value for total deaths and total tests in NZ:
0.9999656651869272
```

[162]:
```
xs6 = who_tests_merged[' New_deaths']
ys6 = who_tests_merged['New tests']
print('Pearson value for new deaths and new tests of all countries:')
print(stats.pearsonr(xs6,ys6)[0])
print('Spearman value for new deaths and new tests of all countries:')
print(stats.spearmanr(xs6,ys6)[0])
```

```
Pearson value for new deaths and new tests of all countries:
0.4895687353190168
Spearman value for new deaths and new tests of all countries:
0.6094366621575511
```

[163]:
```
xus4 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United States']['␣
 ↪New_deaths']
yus4 = who_tests_merged.loc[who_tests_merged[' Country'] == 'United␣
 ↪States']['New tests']
print('Pearson value for new deaths and new tests in the US:')
print(stats.pearsonr(xus4,yus4)[0])
print('Spearman value for new deaths and new tests in the US:')
print(stats.spearmanr(xus4,yus4)[0])
```

```
Pearson value for new deaths and new tests in the US:
0.012489536454235192
Spearman value for new deaths and new tests in the US:
0.1562605549975788
```

[164]:
```
xnz4 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand']['␣
 ↪New_deaths']
ynz4 = who_tests_merged.loc[who_tests_merged[' Country'] == 'New Zealand']['New␣
 ↪tests']
print('Pearson value for new deaths and new tests in NZ:')
print(stats.pearsonr(xnz4,ynz4)[0])
print('Spearman value for new deaths and new tests in NZ:')
```

```
print(stats.spearmanr(xnz4,ynz4)[0])
```

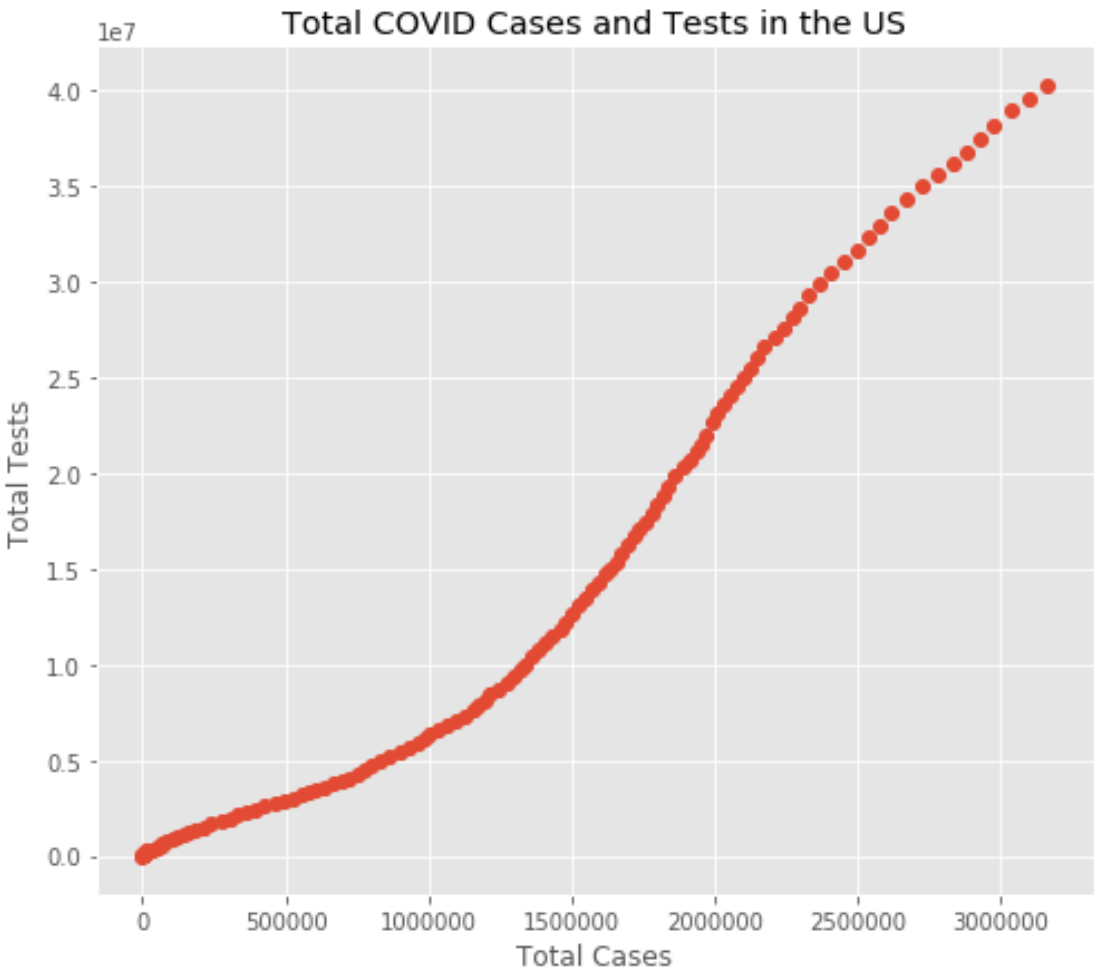
```
Pearson value for new deaths and new tests in NZ:
0.07801255653290888
Spearman value for new deaths and new tests in NZ:
0.15602532849567644
```

[165]:
```
#Create a variable with all of the data for US cases
output_US = who_tests_merged.loc[who_tests_merged[' Country'] == 'United␣
 ↪States'][[' Cumulative_cases', 'Total tests', ' New_cases', 'New tests']]
```

[166]:
```
#Create a variable with all of the data for NZ cases
output_NZ = who_tests_merged.loc[who_tests_merged[' Country'] == 'New␣
 ↪Zealand'][[' Cumulative_cases', 'Total tests', ' New_cases', 'New tests']]
```

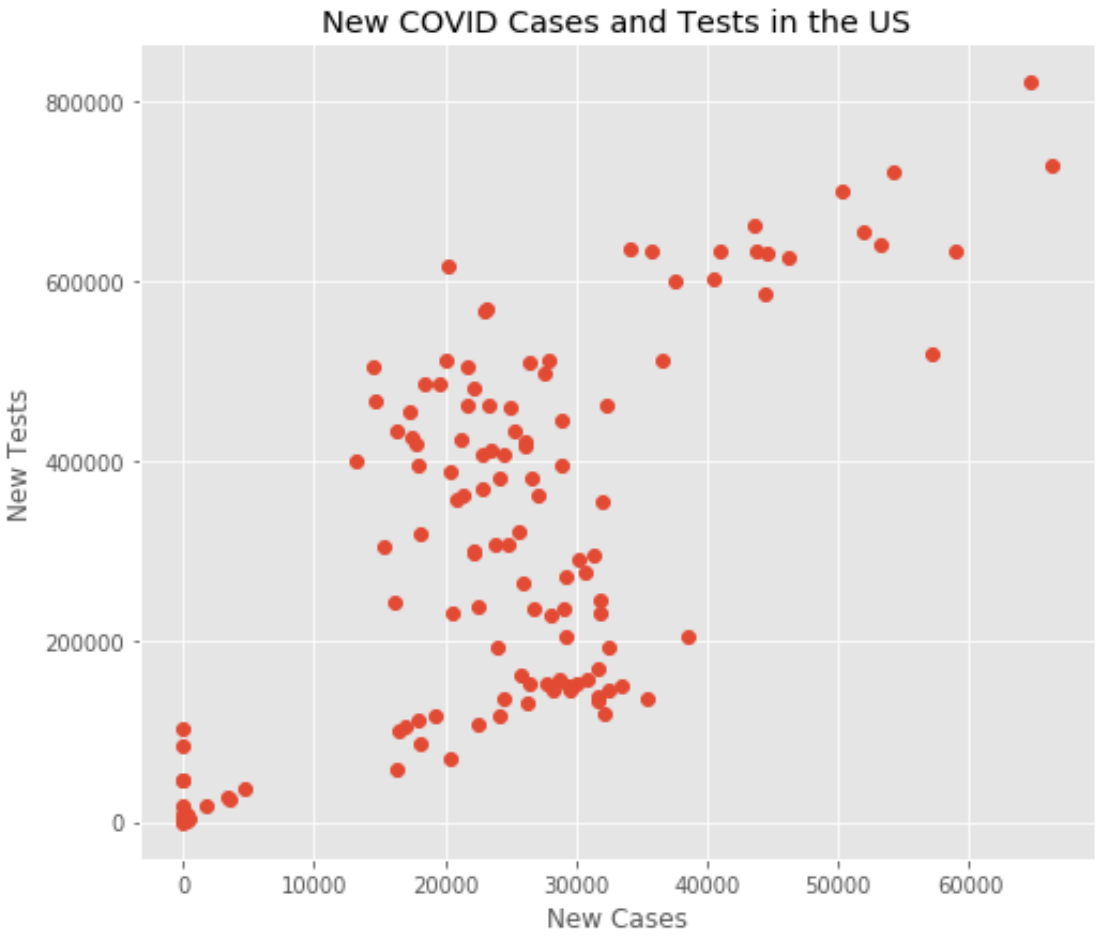
[167]:
```
#Create a plot with the cumulative cases against the total tests in the US
x1 = output_US[' Cumulative_cases']
y1 = output_US['Total tests']
plt.plot(x1, y1,'o')
plt.ylabel('Total Tests')
plt.xlabel('Total Cases')
plt.title('Total COVID Cases and Tests in the US')
```

[167]:
```
Text(0.5, 1.0, 'Total COVID Cases and Tests in the US')
```



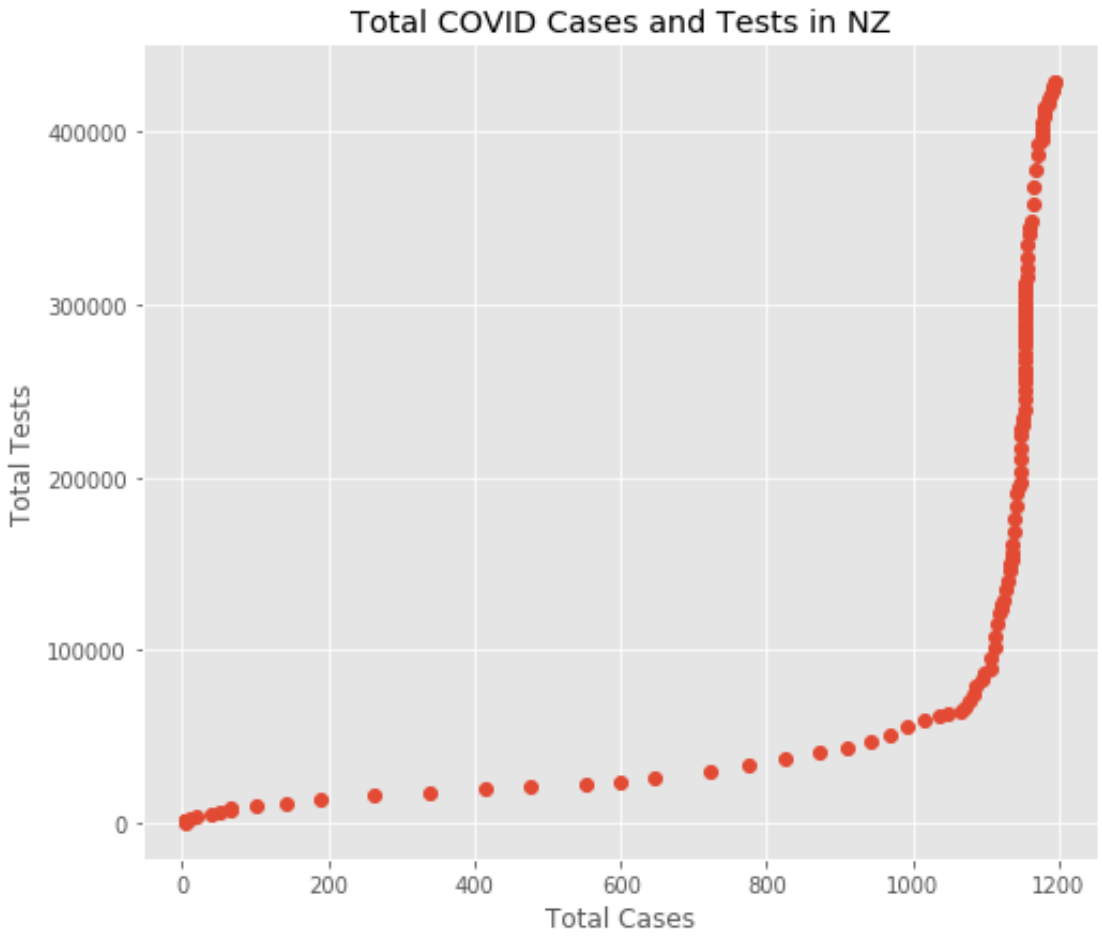
[168]:
```
#Create a plot with the new cases against the new tests in the US
x2 = output_US[' New_cases']
y2 = output_US['New tests']
plt.plot(x2, y2,'o')
plt.ylabel('New Tests')
plt.xlabel('New Cases')
plt.title('New COVID Cases and Tests in the US')
```

[168]:
```
Text(0.5, 1.0, 'New COVID Cases and Tests in the US')
```



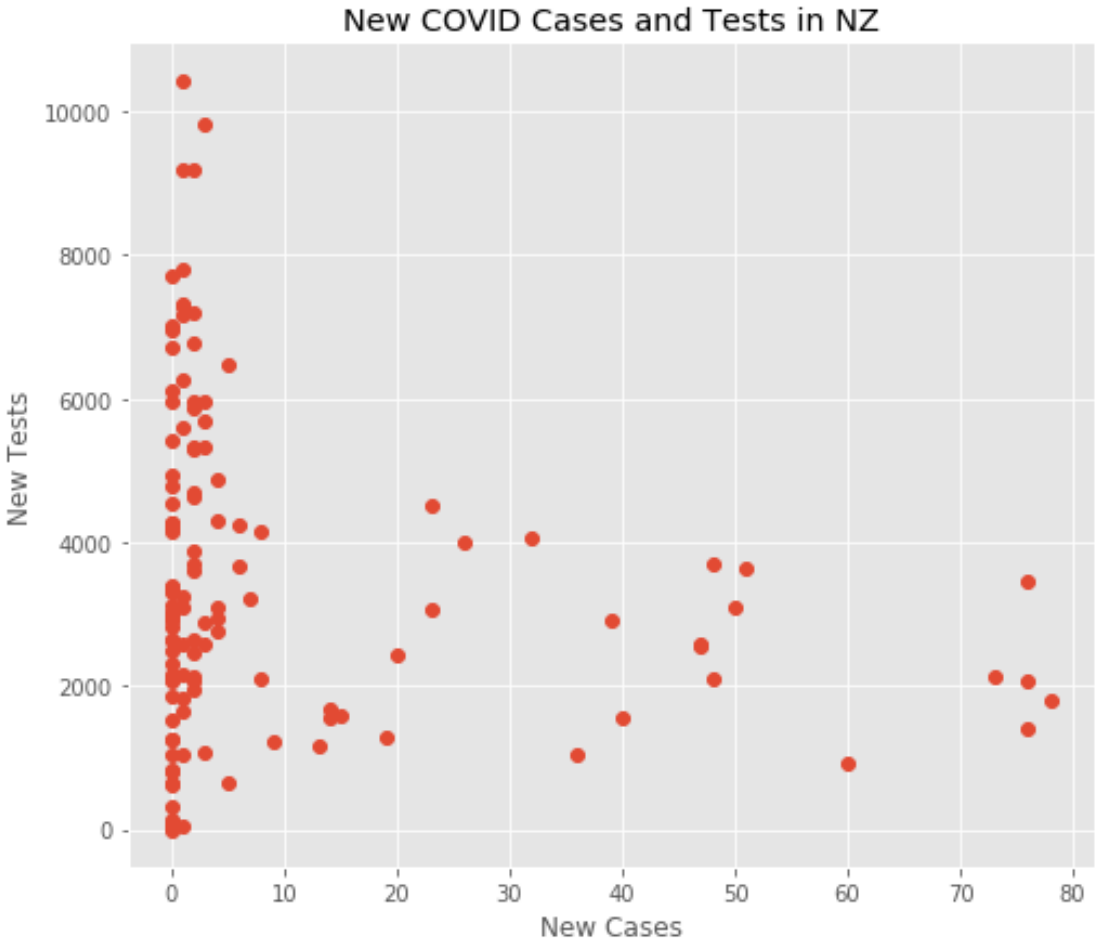
[169]:
```
#Create a plot with the cumulative cases against the total tests in New Zealand
x3 = output_NZ[' Cumulative_cases']
y3 = output_NZ['Total tests']
plt.plot(x3, y3,'o')
plt.ylabel('Total Tests')
plt.xlabel('Total Cases')
plt.title('Total COVID Cases and Tests in NZ')
```

[169]: Text(0.5, 1.0, 'Total COVID Cases and Tests in NZ')



[170]:
```
#Create a plot with the new cases against the new tests in New Zealand
x4 = output_NZ[' New_cases']
y4 = output_NZ['New tests']
plt.plot(x4, y4,'o')
plt.ylabel('New Tests')
plt.xlabel('New Cases')
plt.title('New COVID Cases and Tests in NZ')
```

[170]:
```
Text(0.5, 1.0, 'New COVID Cases and Tests in NZ')
```



5 Government Measures and School Closures

[171]:
```
with open('data/covid_impact_education.csv', newline='') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)
    edu_data=[]
    for row in reader:
        row[0]=datetime.strptime(row[0], '%d/%m/%Y')
        edu_data.append(row)

edu_data=pd.DataFrame(edu_data,columns=['Date', 'ISO', 'Country', 'Scale',␣
 ↪'Note'])
edu_data.head(5)
```

[171]:
```
        Date  ISO   Country      Scale  \
0 2020-02-16  CHN     China  Localized
1 2020-02-16  MNG  Mongolia   National
2 2020-02-17  CHN     China  Localized
3 2020-02-17  MNG  Mongolia   National
4 2020-02-18  CHN     China  Localized

                                                Note
0
1  The government mandated school closures on 27 ...
2
3  The government mandated school closures on 27 ...
4
```

[172]:
```
edu_data.describe()
```

[172]:

	Date	ISO	Country	Scale	Note
count	25546	25546	25546	25546	25546
unique	151	210	211	3	23
top	2020-06-05 00:00:00	CHN	Mongolia	National	
freq	210	151	151	18788	23017
first	2020-02-16 00:00:00	NaN	NaN	NaN	NaN
last	2020-07-15 00:00:00	NaN	NaN	NaN	NaN

The dataset for global school closure contains 24456 pieces of school closure/reopen records in 211 countries from 2020-02-16 to 2020-07-15. This dataset contains five features: dates of data aquisition, ISO codes for countries, country names, scale of measure(localized/national), and optional notes. There are five different school measures in total: national closure, localized closure, national reopen, localized reopen, and open.

[173]:
```
edu_data=edu_data.sort_values(['ISO','Date'])
```

[174]:
```
with open('WHO-COVID-19-global-data.csv', newline='') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)
    who_data=[]
    for row in reader:
        row[0]=datetime.strptime(row[0], '%Y-%m-%d')
        who_data.append(row)

who_data=pd.
 ↪DataFrame(who_data,columns=['Date_reported','Country_code','Country','WHO_region','New_cases'
who_data.head(5)
```

[174]:

	Date_reported	Country_code	Country	WHO_region	New_cases
0	2020-02-24	AF	Afghanistan	EMRO	1
1	2020-02-25	AF	Afghanistan	EMRO	0

```
2    2020-02-26           AF  Afghanistan       EMRO         0
3    2020-02-27           AF  Afghanistan       EMRO         0
4    2020-02-28           AF  Afghanistan       EMRO         0

  Cumulative_cases New_deaths Cumulative_deaths
0                1          0                 0
1                1          0                 0
2                1          0                 0
3                1          0                 0
4                1          0                 0
```

[175]:
```
who_data.describe()
```

[175]:
```
              Date_reported Country_code Country WHO_region New_cases  \
count                 27772        27772   27772      27772     27772
unique                  184          216     216          7      2541
top     2020-05-23 00:00:00           CN   China       EURO         0
freq                    216          184     184       8235      9752
first   2020-01-11 00:00:00          NaN     NaN        NaN       NaN
last    2020-07-12 00:00:00          NaN     NaN        NaN       NaN

       Cumulative_cases New_deaths Cumulative_deaths
count             27772      27772             27772
unique             9258        631              3110
top                   1          0                 0
freq                969      18160              7526
first               NaN        NaN               NaN
last                NaN        NaN               NaN
```

The dataset of global cases from WHO contains 27772 pieces of case/death data in 216 countries from 2020-01-11 to 2020-07-12. This dataset contains date reported, ISO country code, WHO region, country name, new cases, cumulative cases, new deaths, cumulative deaths.

[176]:
```
with open('data/acaps_covid19_government_measures_dataset_0.csv',␣
 ↪newline='\n',encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)
    gov_data=[]
    for row in reader:
        row[-2]=datetime.strptime(row[-2], '%Y-%m-%d')
        gov_data.append(row)

gov_data=pd.
 ↪DataFrame(gov_data,columns=['ID','COUNTRY','ISO','ADMIN_LEVEL_NAME','PCODE','REGION','LOG_TYP
 ↪source'])
gov_data.head(5)
```

[176]:

	ID	COUNTRY	ISO	ADMIN_LEVEL_NAME	PCODE	REGION
0	1	Afghanistan	AFG			Asia
1	2	Afghanistan	AFG	Kabul		Asia
2	3	Afghanistan	AFG			Asia
3	4	Afghanistan	AFG			Asia
4	5	Afghanistan	AFG			Asia

	LOG_TYPE	CATEGORY
0	Introduction / extension of measures	Public health measures
1	Introduction / extension of measures	Public health measures
2	Introduction / extension of measures	Public health measures
3	Introduction / extension of measures	Governance and socio-economic measures
4	Introduction / extension of measures	Social distancing

	MEASURE	TARGETED_POP_GROUP
0	Health screenings in airports and border cross...	No
1	Isolation and quarantine policies	No
2	Awareness campaigns	No
3	Emergency administrative structures activated ...	No
4	Limit public gatherings	No

	COMMENTS	NON_COMPLIANCE	DATE_IMPLEMENTED
0			2020-02-12
1			2020-02-12
2			2020-02-12
3			2020-02-12
4	Nevruz festival cancelled		2020-03-12

	SOURCE	SOURCE_TYPE
0	Ministry of Health	Government
1	Ministry of Health	Government
2	Ministry of Health	Government
3	Ministry of Health	Government
4	AA	Media

	LINK	ENTRY_DATE
0	https://moph.gov.af/en/moph-held-emergency-mee...	2020-03-14
1	https://moph.gov.af/en/moph-held-emergency-mee...	2020-03-14
2	https://moph.gov.af/en/moph-held-emergency-mee...	2020-03-14
3	https://moph.gov.af/en/moph-held-emergency-mee...	2020-03-14

```
4  https://www.aa.com.tr/en/asia-pacific/coronavi... 2020-03-14

  Alternative source
0
1
2
3
4
```

[177]:
```
gov_data.describe()
```

[177]:
```
           ID      COUNTRY    ISO ADMIN_LEVEL_NAME  PCODE  REGION  \
count   14852        14852  14852             14852  14852   14852
unique  14852          194    193               777      1       6
top     10037  Philippines    PHL                          Europe
freq        1          356    356             13188  14852    4619
first     NaN          NaN    NaN               NaN    NaN     NaN
last      NaN          NaN    NaN               NaN    NaN     NaN

                                    LOG_TYPE                CATEGORY  \
count                                14852                   14852
unique                                   2                       6
top     Introduction / extension of measures  Public health measures
freq                                 12121                    4690
first                                  NaN                     NaN
last                                   NaN                     NaN

                  MEASURE TARGETED_POP_GROUP COMMENTS  NON_COMPLIANCE  \
count               14852              14852    14852           14852
unique                 42                  4    14432              13
top     Economic measures                 No           Not applicable
freq                 2047               8911      149           10565
first                 NaN                NaN      NaN             NaN
last                  NaN                NaN      NaN             NaN

       DATE_IMPLEMENTED      SOURCE SOURCE_TYPE  \
count             14852       14852       14852
unique              198        1422           8
top          2020-03-16  Government  Government
freq                339        1310        9849
first               NaN         NaN         NaN
last                NaN         NaN         NaN

                                                  LINK  \
count                                            14852
unique                                            8724
top     https://pandemic.internationalsos.com/2019-nco...
```

```
freq                                                    431
first                                                   NaN
last                                                    NaN

                ENTRY_DATE Alternative source
count                14852              14852
unique                 104                845
top    2020-04-21 00:00:00
freq                   492              13720
first  2020-03-04 00:00:00                NaN
last   2020-07-09 00:00:00                NaN
```

[178]:
```
gov_data=gov_data.sort_values(['ISO','ENTRY_DATE'])
```

- Categories of measures: Governance and socio-economic measures, Humanitarian exemption,Lockdown,Movement restrictions,Public health measures,Social distancing
- More detailed measures in measure column.
- Date implemented column for the effective date of the measure.
- Non Compliance column for punishment of this measure.

[179]:
```
iso_dict={}
f = open("data/countrycode.txt", "r")
for x in f:
    words=x.split("        ")
    if len(words)<5:
        continue
    else:
        iso_dict[words[0]]=(words[1],words[4])
```

[180]:
```
who_data=who_data.sort_values(['Country_code','Date_reported'])
who_data=np.array(who_data)
```

[181]:
```
who_dict={}
u, indices = np.unique(who_data[:,1], return_index=True)
np.append(indices,[len(who_data)])
for i in range(len(indices)-1):
    key=who_data[indices[i]][1]
    who_dict[key]={}
    who_dict[key]['Date_reported']=who_data[indices[i]:indices[i+1],0]
    who_dict[key]['New_cases']=np.array( [int(i) for i in who_data[indices[i]:
 ↪indices[i+1],4]])
    who_dict[key]['Cumulative_cases']=np.array( [int(i) for i in␣
 ↪who_data[indices[i]:indices[i+1],5]])
    who_dict[key]['New_deaths']=np.array( [int(i) for i in who_data[indices[i]:
 ↪indices[i+1],6]])
    who_dict[key]['Cumulative_deaths']=np.array( [int(i) for i in␣
 ↪who_data[indices[i]:indices[i+1],7]])
```

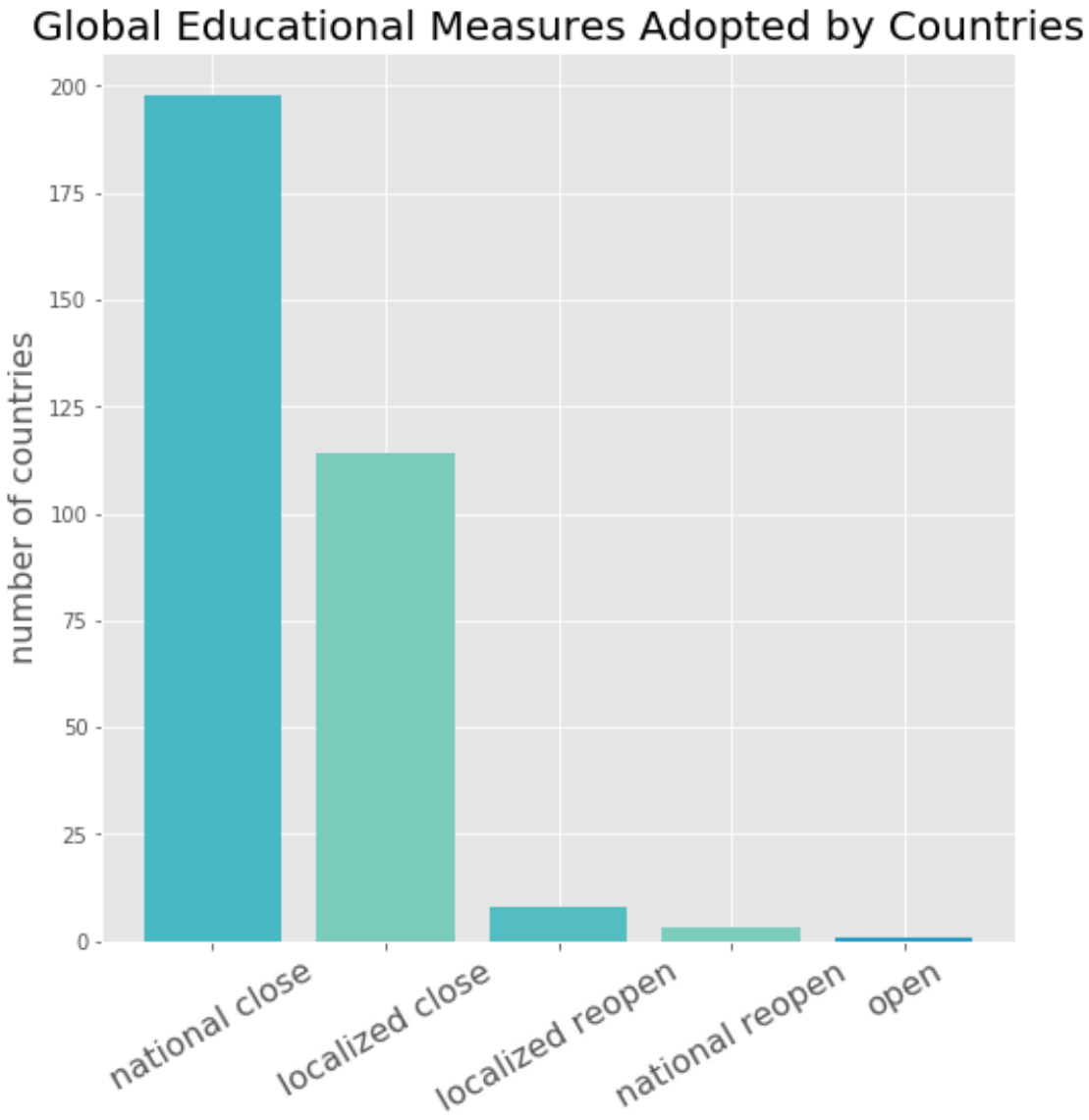
[182]:
```
# reorganize data by country/measure, only record ealiest pushing date
edu_data=np.array(edu_data)
edu_by_country={}
edu_by_measure={"national close":{},"localized close":{},"localized reopen":
 ↪{},"national reopen":{},"open":{}}
for datum in edu_data:
    if datum[1] not in edu_by_country.keys():
        edu_by_country[datum[1]]={}
    if(datum[3]=="National"):
        if len(datum[4])>0 and ("open" in datum[4]):
            if "national close" in edu_by_country[datum[1]].keys() or "localized␣
 ↪close" in edu_by_country[datum[1]].keys():
                if "national reopen" not in edu_by_country[datum[1]].keys():
                    edu_by_country[datum[1]]["national reopen"]=datum[0]
                    edu_by_measure["national reopen"][datum[1]]=datum[0]
            else:
                if "open" not in edu_by_country[datum[1]].keys():
                    edu_by_country[datum[1]]["open"]=datum[0]
                    edu_by_measure["open"][datum[1]]=datum[0]
        else:
            if "national close" not in edu_by_country[datum[1]].keys():
                edu_by_country[datum[1]]["national close"]=datum[0]
                edu_by_measure["national close"][datum[1]]=datum[0]
    else:
        if len(datum[4])>0 and ("open" in datum[4]):
            if "national close" in edu_by_country[datum[1]].keys() or "localized␣
 ↪close" in edu_by_country[datum[1]].keys():
                if "localized reopen" not in edu_by_country[datum[1]].keys():
                    edu_by_country[datum[1]]["localized reopen"]=datum[0]
                    edu_by_measure["localized reopen"][datum[1]]=datum[0]
            else:
                 if "open" not in edu_by_country[datum[1]].keys():
                    edu_by_country[datum[1]]["open"]=datum[0]
                    edu_by_measure["open"][datum[1]]=datum[0]
        else:
            if "localized close" not in edu_by_country[datum[1]].keys():
                edu_by_country[datum[1]]["localized close"]=datum[0]
                edu_by_measure["localized close"][datum[1]]=datum[0]
```

5.0.1 Education Measures Adopted by Countries

[183]:
```
measure_count=[]
tags=tuple(edu_by_measure.keys())
for key in tags:
    measure_count.append(len(edu_by_measure[key].keys()))
```

```
ind = np.arange(len(tags))
plt.figure(figsize=(8,8))
my_cmap = cm.get_cmap('YlGnBu')
color=[random.uniform(0.1, 0.6) for i in range(len(tags))]
print(measure_count)
p1 = plt.bar(ind, measure_count,color=my_cmap(color))
plt.ylabel('number of countries',fontsize=16)
plt.title('Global Educational Measures Adopted by Countries',fontsize=20)
plt.xticks(ind, tags)
plt.xticks(rotation=30,fontsize=16)
plt.show()
```

```
[198, 114, 8, 3, 1]
```



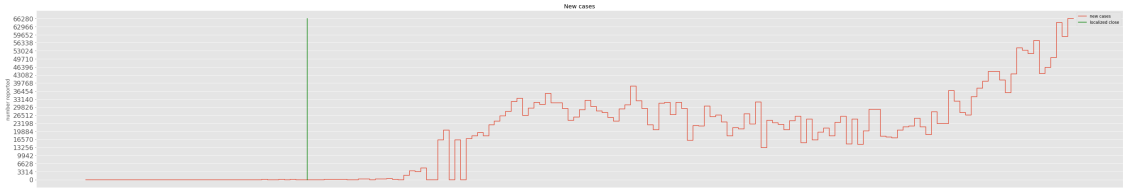
5.0.2 visualization of cases/deaths curves and educational measures

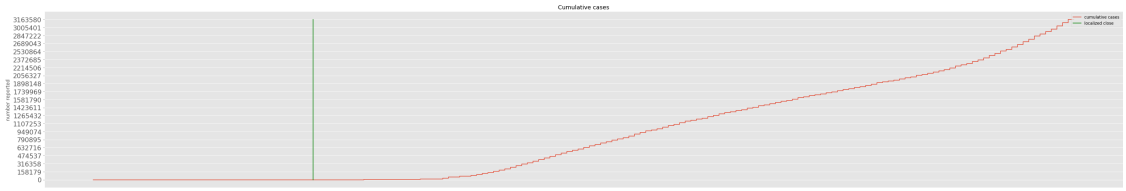
[184]:
```
def draw_edu_graph_for_country(country_iso):
    date= who_dict[country_iso]['Date_reported']
    ind = np.arange(len(who_dict[country_iso]['Date_reported']))
    series_tag=['New_cases','Cumulative_cases','New_deaths','Cumulative_deaths']
    for i in range(len(series_tag)):
        legend_line=[]
        legend_tag=[]
        plt.figure(figsize=(48,8))
        series=series_tag[i]
        maxv=who_dict[country_iso][series].max()
        data=who_dict[country_iso][series]
        plt.step(ind, data,label=series.replace("_"," ").lower())
        for key in edu_by_country[iso_dict[country_iso][0]]:
            if key=='national close':
                line='r-'
            elif key == "localized close":
                line='g-'
            elif key=='national reopen':
                line='black'
            elif key == "localized reopen":
                line='c-'
            else:
                line='m-'
            x=[(edu_by_country[iso_dict[country_iso][0]][key]-date[0]).days for␣
↪i in range(21)]
            y=[i for i in range(0,(maxv//10+1)*10, maxv//20)][:21]
            plt.plot(x,y,line,label=key)
        plt.ylabel('number reported')
        plt.title(series.replace("_"," "))
        if i==len(series_tag)-1:
            plt.xticks(ind,date)
            plt.xticks(rotation=90,fontsize=16)
        else:
            plt.xticks([], [])

        handles, labels = plt.gca().get_legend_handles_labels()
        for h,l in zip(handles,labels):
            if l not in legend_tag:
                legend_line.append(h)
                legend_tag.append(l)
        plt.yticks(np.arange(0, (maxv//10+1)*10, step=maxv//20),fontsize=16)
```

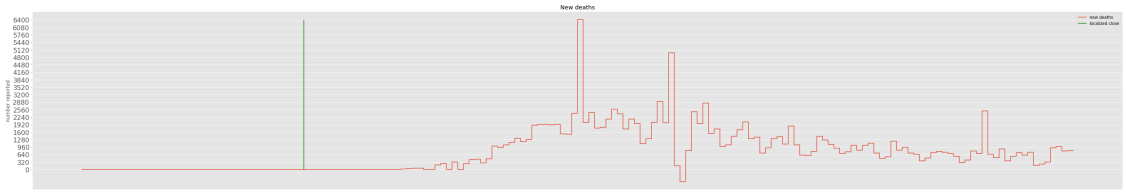
```
        plt.legend(legend_line,legend_tag,loc='upper right',fontsize=10)
        plt.show()
```

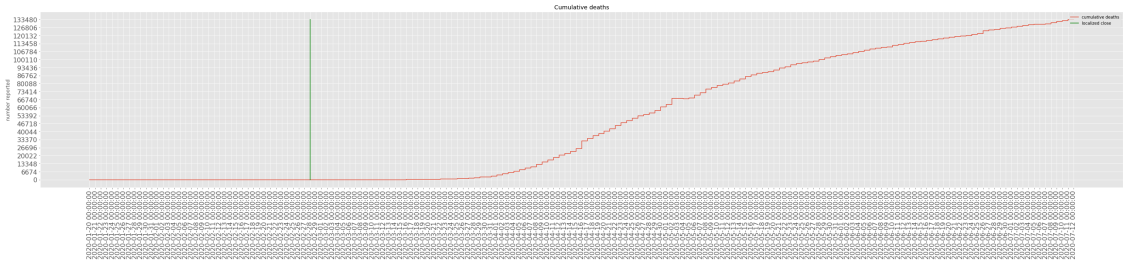
[185]:
```
country_list=["US","CN","IT","BR","IN","RU","ZA"]
for iso in country_list:
    print(iso_dict[iso][1])
    draw_edu_graph_for_country(iso)
    print("=====================================================")
```

```
United States
```



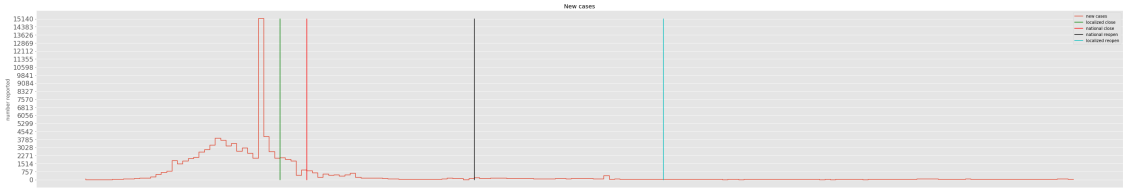


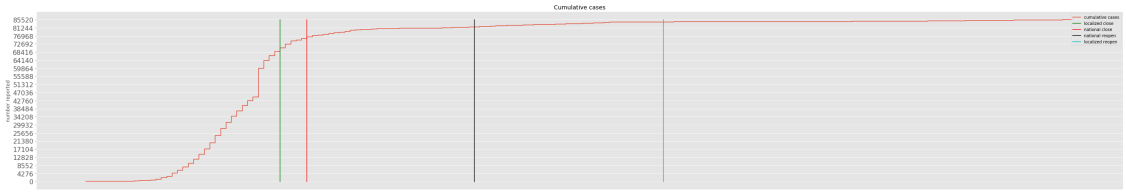


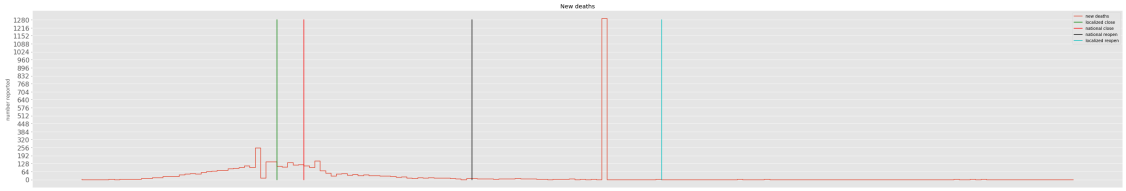


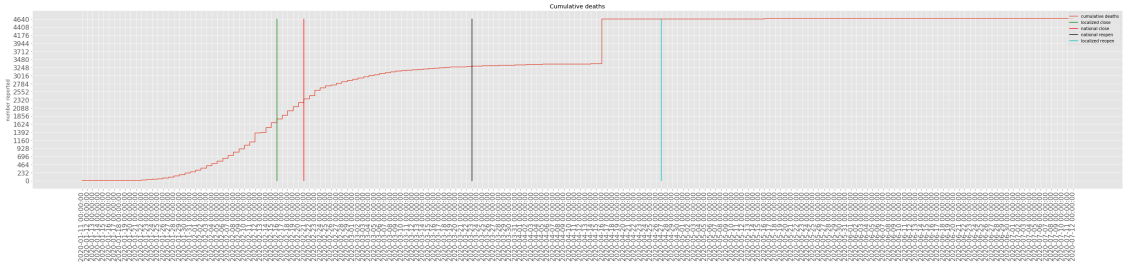

```
=====================================================
```

China


New cases
number reported
new cases
localized close
national close
national reopen
localized reopen

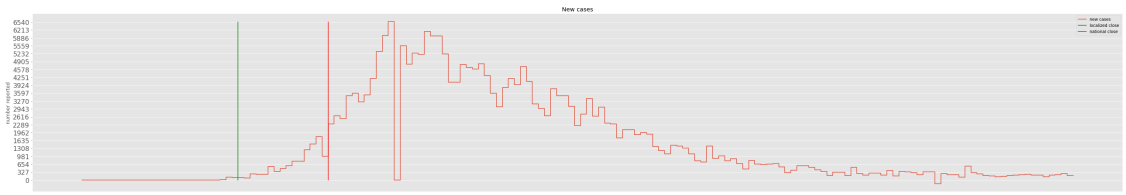

Cumulative cases
number reported
cumulative cases
localized close
national close
national reopen
localized reopen

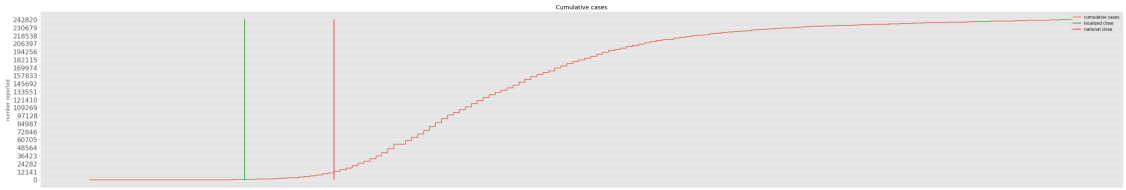

New deaths
number reported
new deaths
localized close
national close
national reopen
localized reopen

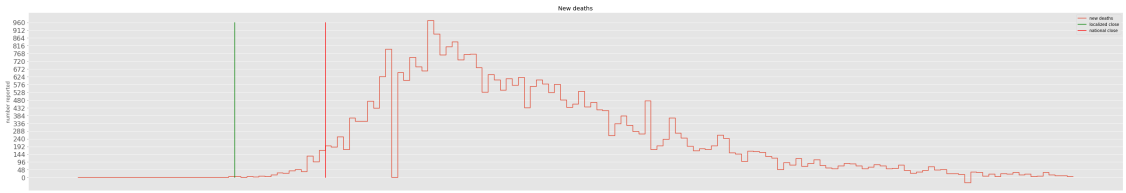

Cumulative deaths
number reported
cumulative deaths
localized close
national close
national reopen
localized reopen

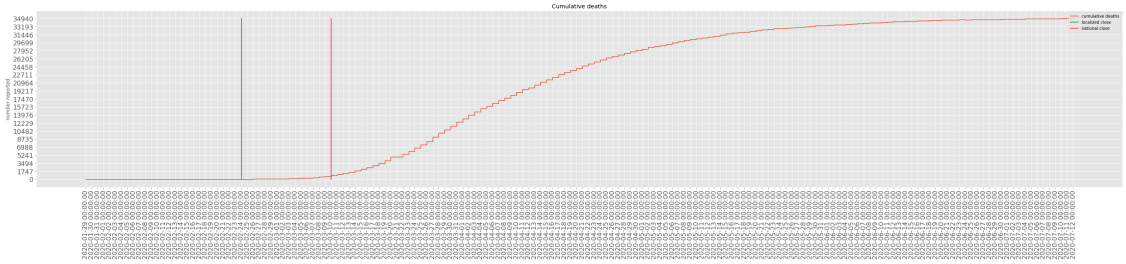
==========

Italy


New cases
number reported
new cases
localized close
national close

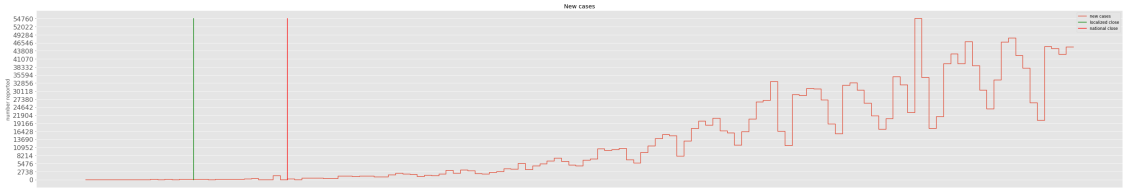

Cumulative cases
number reported
cumulative cases
localized close
national close

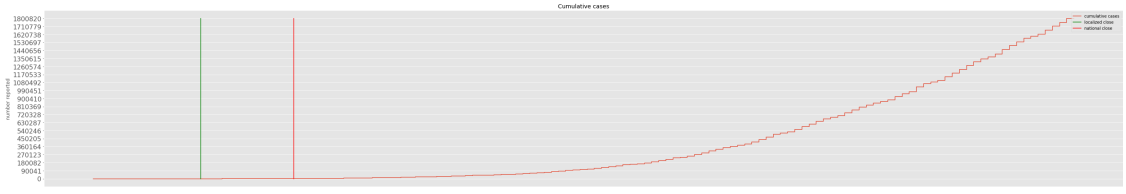

New deaths
number reported
new deaths
localized close
national close

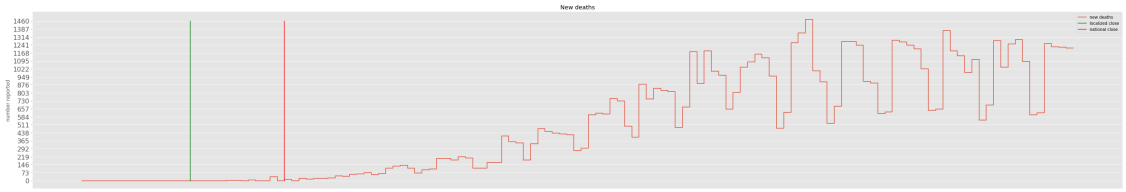

Cumulative deaths
number reported
cumulative deaths
localized close
national close

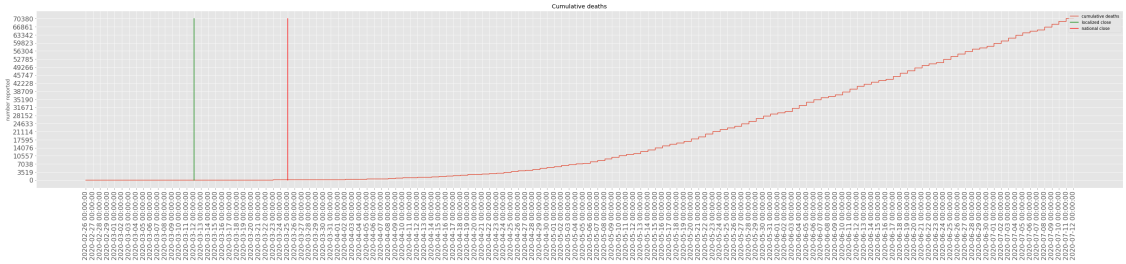
==========

Brazil


New cases
number reported
new cases
localized close
national close

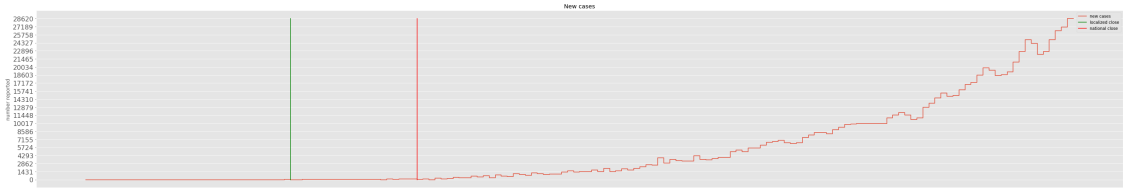

Cumulative cases
number reported
cumulative cases
localized close
national close

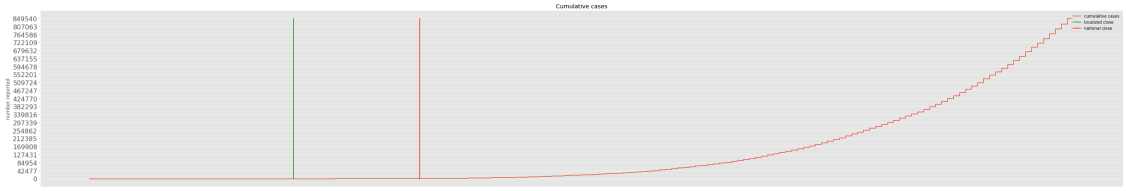

New deaths
number reported
new deaths
localized close
national close

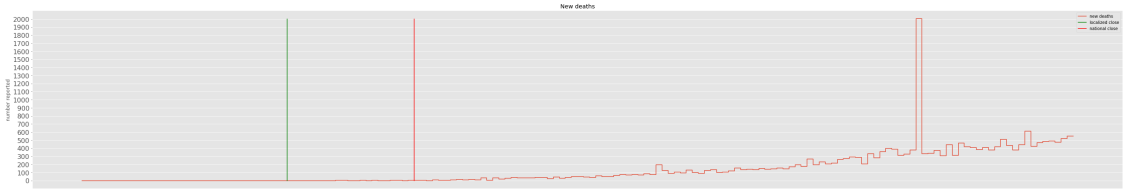

Cumulative deaths
number reported
cumulative deaths
localized close
national close

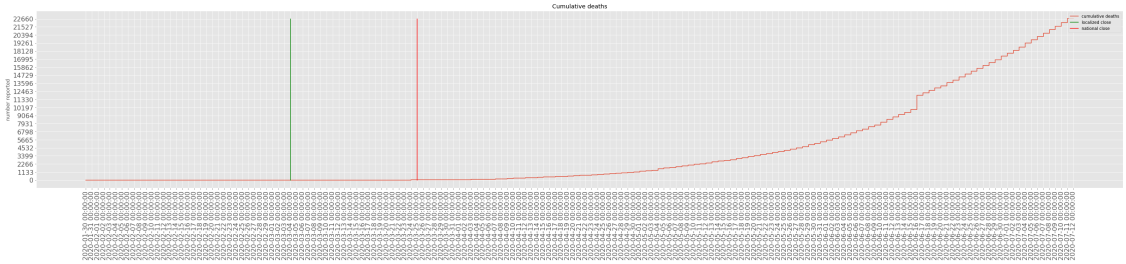
==========

India


New cases
number reported
new cases
localized close
national close

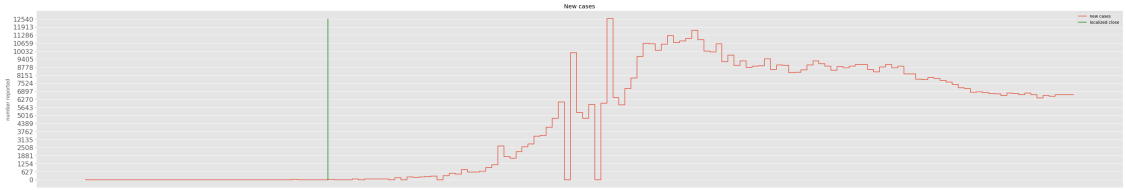

Cumulative cases
number reported
cumulative cases
localized close
national close

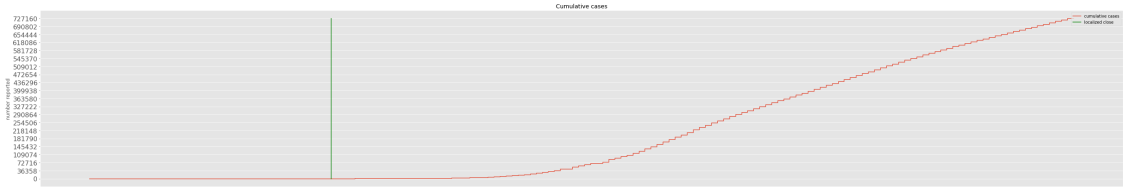

New deaths
number reported
new deaths
localized close
national close

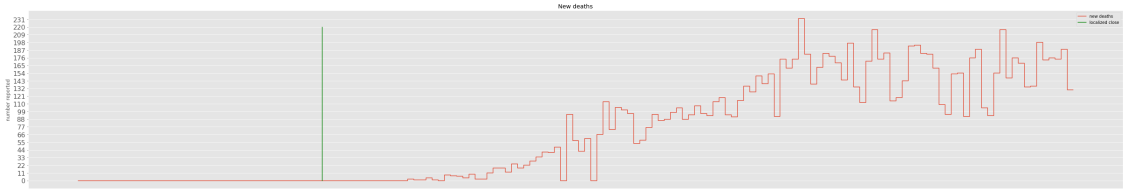

Cumulative deaths
number reported
cumulative deaths
localized close
national close

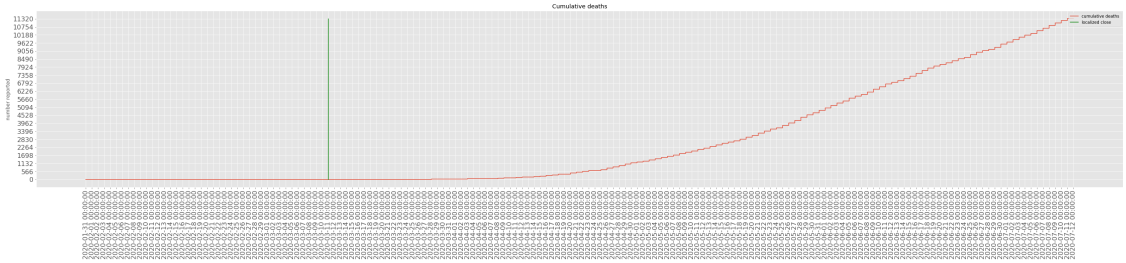
==========

Russia


New cases
number reported
new cases
localized close

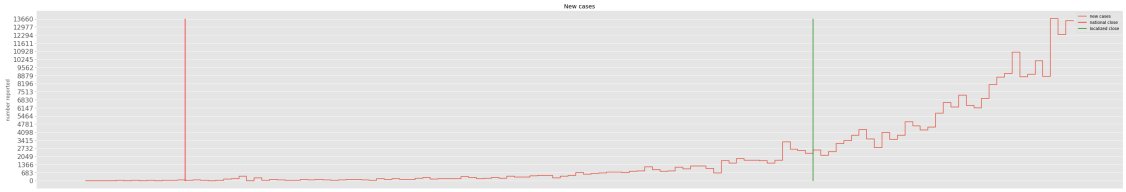

Cumulative cases
number reported
cumulative cases
localized close

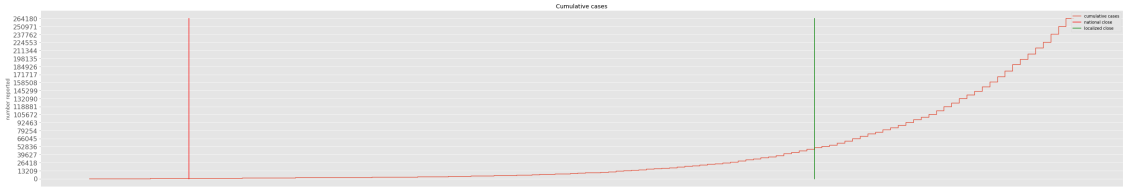

New deaths
number reported
new deaths
localized close

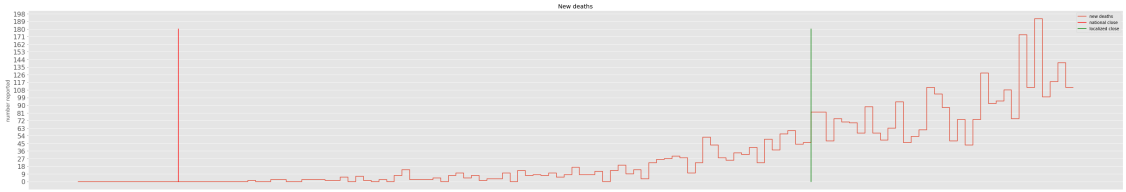

Cumulative deaths
number reported
cumulative deaths
localized close

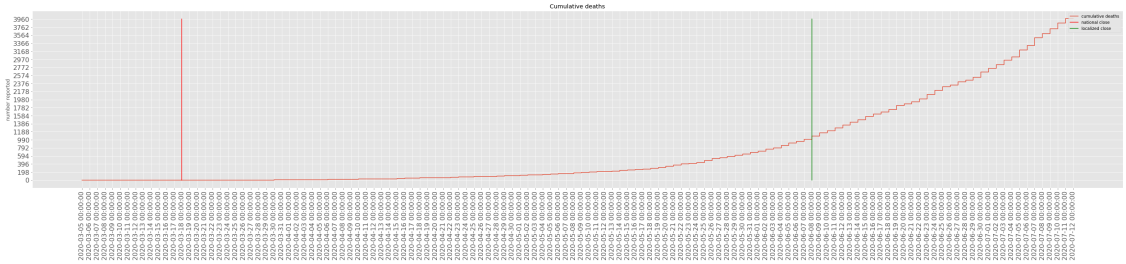
==========

South Africa









==========

[186]:
```
gov_data=np.array(gov_data)
```

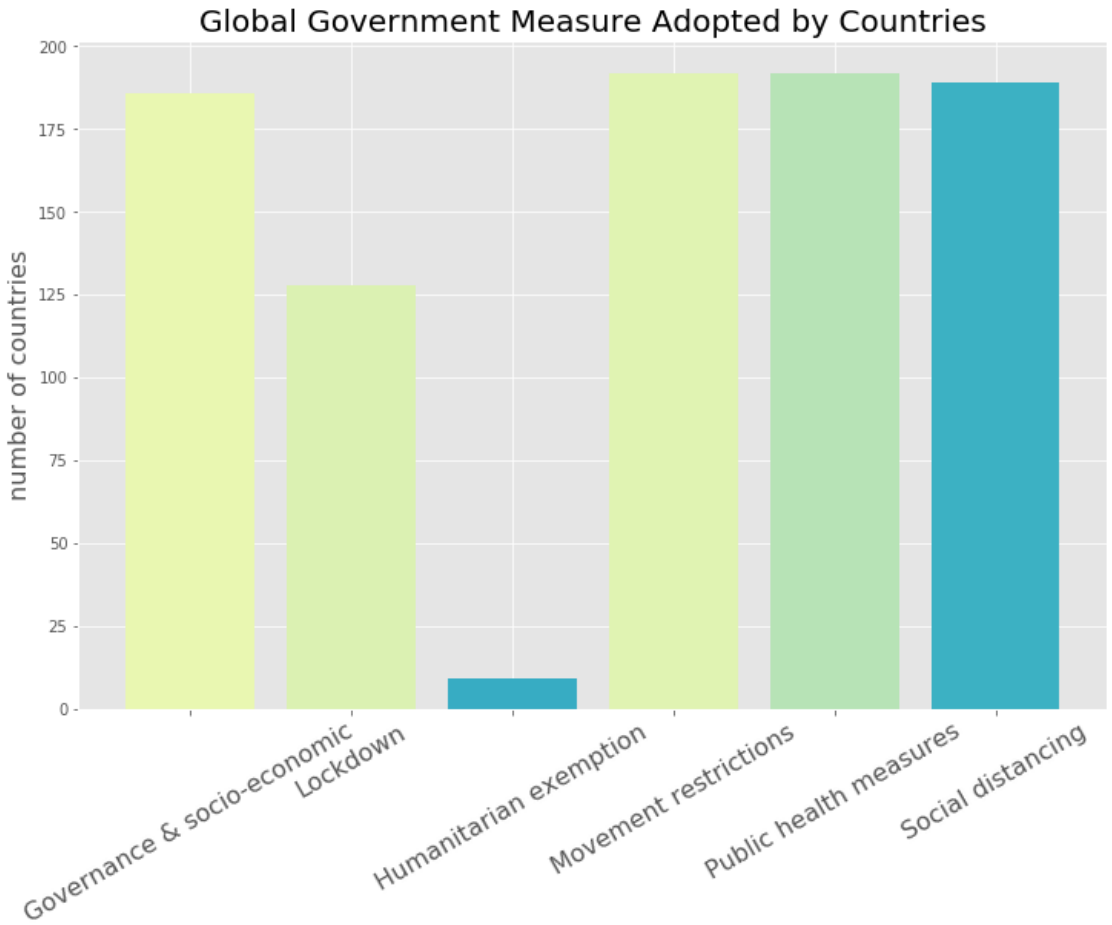
[187]:
```
#ID','COUNTRY','ISO','ADMIN_LEVEL_NAME','PCODE','REGION','LOG_TYPE','CATEGORY','MEASURE','TARGE
 ↪source'])
#preserve only earliest record if a measure, indexed by country and category
gov_by_country={}
```

```
gov_by_measure={"Governance and socio-economic measures":{}, "Lockdown":
 ↪{},"Humanitarian exemption":{},"Movement restrictions":{},"Public health␣
 ↪measures":{},"Social distancing":{}}
for datum in gov_data:
    if datum[3]:
        continue #only count country level
    if datum[2] not in gov_by_country.keys():
        gov_by_country[datum[2]]={}
    if datum[7] not in gov_by_country[datum[2]].keys():
        gov_by_country[datum[2]][datum[7]]={}
    if datum[2] not in gov_by_measure[datum[7]].keys():
        gov_by_measure[datum[7]][datum[2]]={}
    if datum[8] not in gov_by_country[datum[2]][datum[7]].keys():
        date= datum[-2]
        if datum[-6]!="":
            date=min(datetime.strptime(datum[-6], '%Y-%m-%d'),date)
        gov_by_country[datum[2]][datum[7]][datum[8].
 ↪strip("\xa0")]=[date,datum[-7]]
        gov_by_measure[datum[7]][datum[2]][datum[8].
 ↪strip("\xa0")]=[date,datum[-7]]
```

5.0.3 government measures adopted by countries

[188]:
```
measure_count=[]
keyset=list(gov_by_measure.keys())
keyset[keyset.index('Governance and socio-economic measures')] = 'Governance &␣
 ↪socio-economic'
tags=tuple(gov_by_measure.keys())
for key in tags:
    measure_count.append(len(gov_by_measure[key].keys()))
ind = np.arange(len(tags))
plt.figure(figsize=(12,8))
my_cmap = cm.get_cmap('YlGnBu')
print(measure_count)
color=[random.uniform(0.1, 0.65) for i in range(len(tags))]
p1 = plt.bar(ind, measure_count,color=my_cmap(color))
plt.ylabel('number of countries',fontsize=16)
plt.title('Global Government Measure Adopted by Countries',fontsize=20)
plt.xticks(ind, tuple(keyset))
plt.xticks(rotation=30,fontsize=16)
plt.show()
```

```
[186, 128, 9, 192, 192, 189]
```



5.0.4 visualization of cases/deaths curves and educational measures

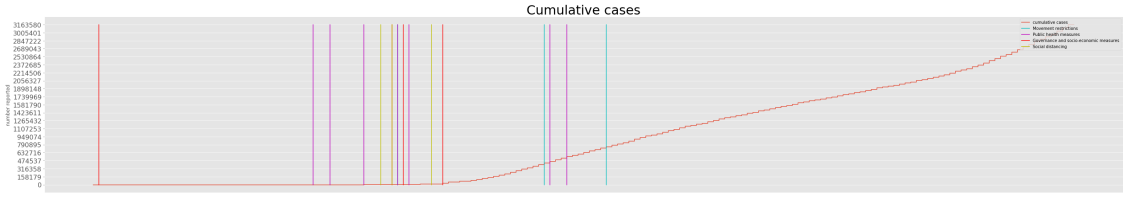
[189]:
```
def draw_gov_graph_for_country(iso):
    date= who_dict[iso]['Date_reported']
    ind = np.arange(len(who_dict[iso]['Date_reported']))
    series_tag=["Governance and socio-economic measures",␣
 ↪"Lockdown","Humanitarian exemption","Movement restrictions","Public health␣
 ↪measures","Social distancing"]
    series_tag=['New_cases','Cumulative_cases','New_deaths','Cumulative_deaths']

    for i in range(len(series_tag)):
        legend_line=[]
        legend_tag=[]
        plt.figure(figsize=(48,8))
        series=series_tag[i]
        maxv=who_dict[iso][series].max()
        data=who_dict[iso][series]
```

```
        plt.step(ind, data,label=series.replace("_"," ").lower())
        for key in gov_by_country[iso_dict[iso][0]]:
            if key=='Governance and socio-economic measures':
                line='r-'
            elif key == "Lockdown":
                line='g-'
            elif key=='Humanitarian exemption':
                line='black'
            elif key == "Movement restrictions":
                line='c-'
            elif key== 'Public health measures':
                line='m-'
            else:
                line='y-'
            for sub in gov_by_country[iso_dict[iso][0]][key].keys():
                x=[(gov_by_country[iso_dict[iso][0]][key][sub][0]-date[0]).days␣
↪for i in range(21)]
                y=[i for i in range(0,(maxv//10+1)*10, maxv//20)][:21]
                plt.plot(x,y,line,label=key)

        plt.ylabel('number reported')
        plt.title(series.replace("_"," "),fontsize=32)
        if i==len(series_tag)-1:
            plt.xticks(ind,date)
            plt.xticks(rotation=90,fontsize=16)
        else:
            plt.xticks([], [])

        handles, labels = plt.gca().get_legend_handles_labels()
        for h,l in zip(handles,labels):
            if l not in legend_tag:
                legend_line.append(h)
                legend_tag.append(l)
        plt.yticks(np.arange(0, (maxv//10+1)*10, step=maxv//20),fontsize=16)
        plt.legend(legend_line,legend_tag,loc='upper right',fontsize=10)
        plt.show()
```

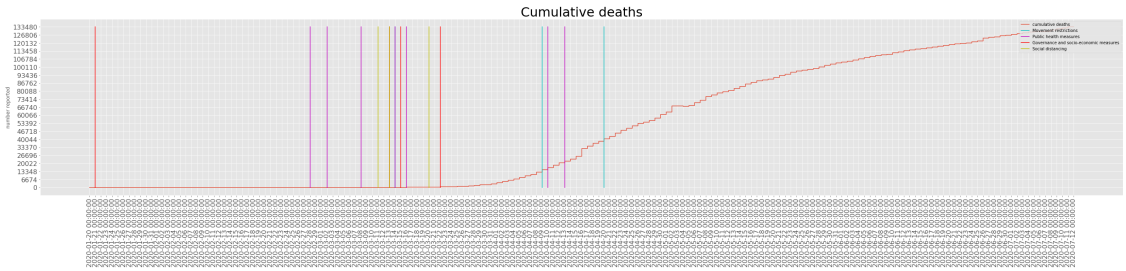
[190]:
```
country_list=["US","CN","IT","BR","IN","RU","ZA"]
for iso in country_list:
    print(iso_dict[iso][1])
    draw_gov_graph_for_country(iso)
    print("=====================================================")
```

```
United States
```


New cases


Cumulative cases

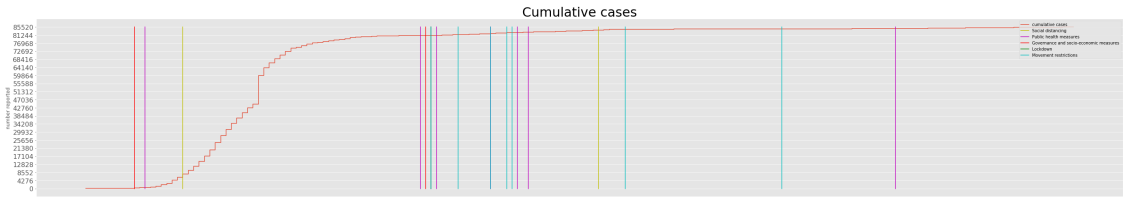

New deaths

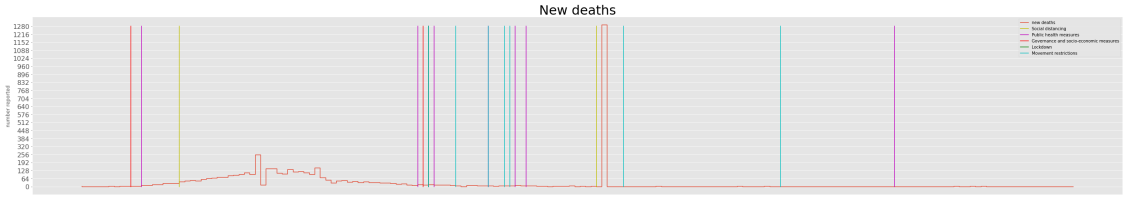

Cumulative deaths

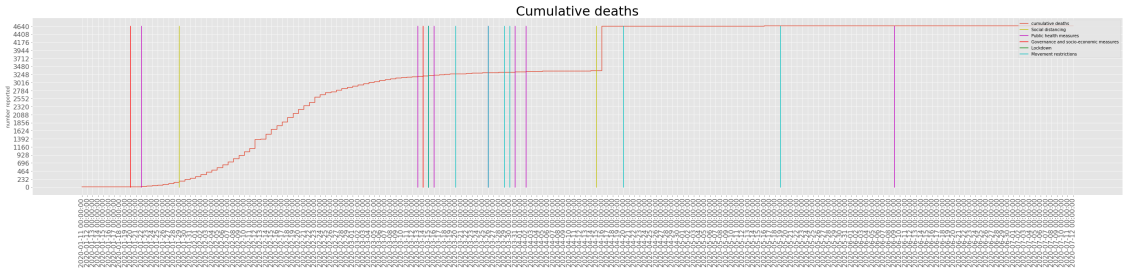
==========

China


New cases

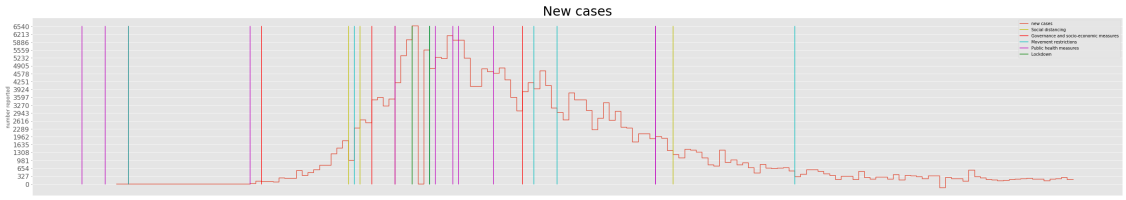

Cumulative cases

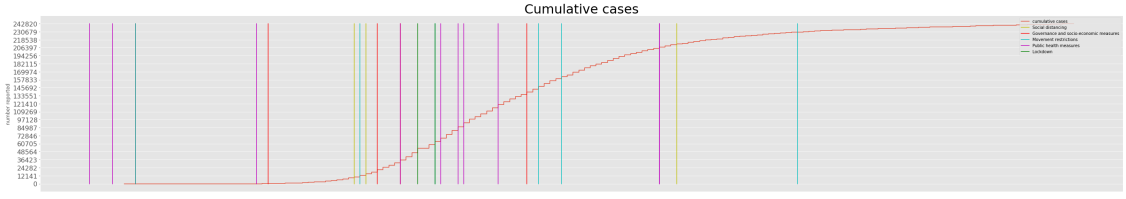

New deaths

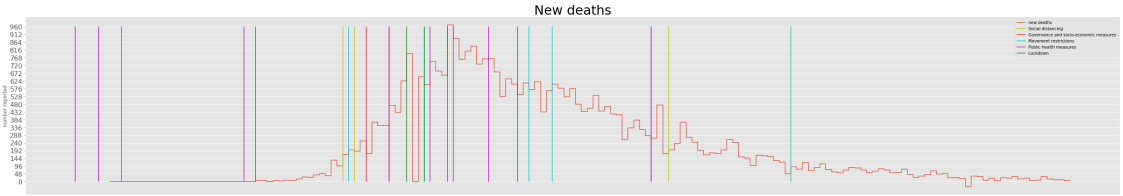

Cumulative deaths

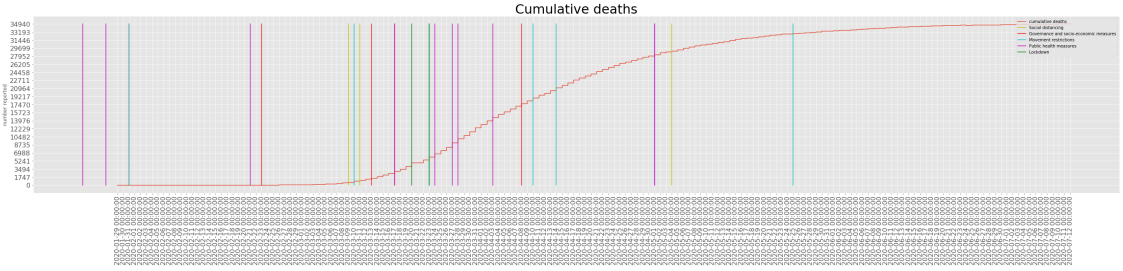
==========

Italy


New cases

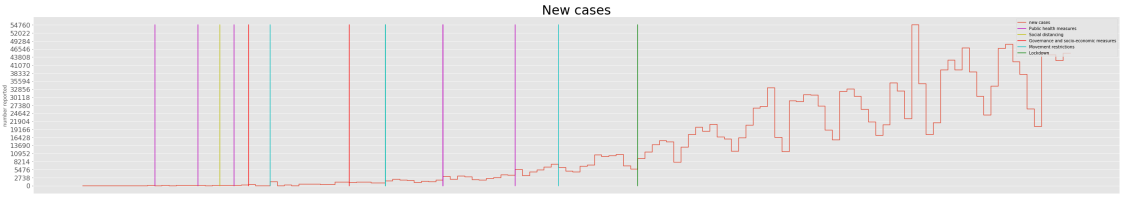

Cumulative cases

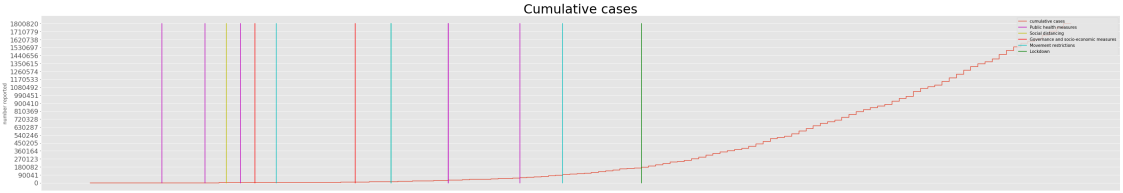

New deaths

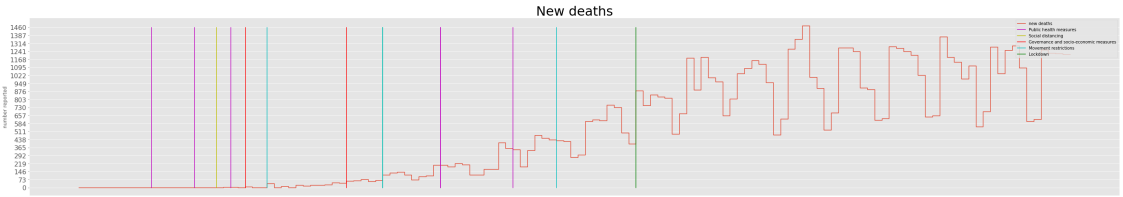

Cumulative deaths

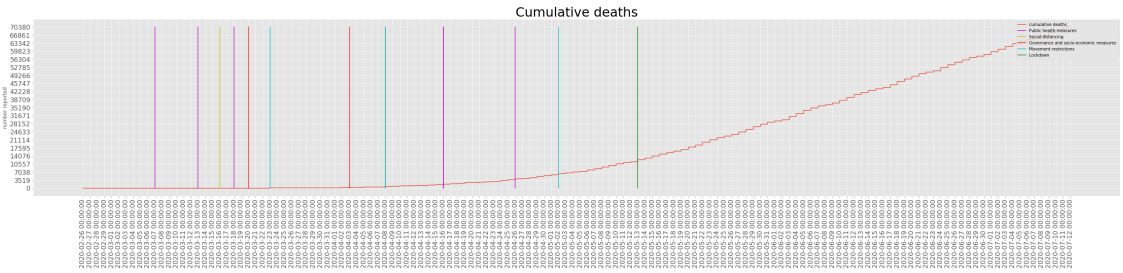
==========

Brazil


New cases

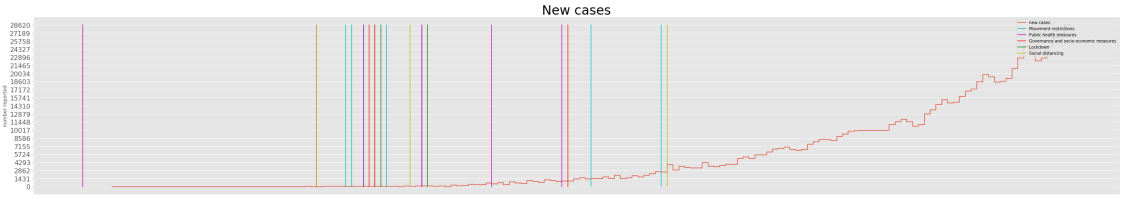

Cumulative cases

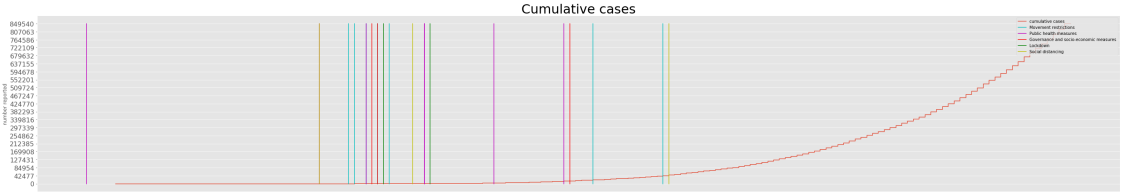

New deaths

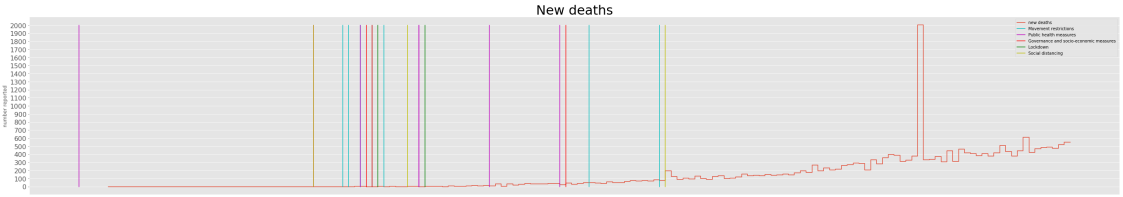

Cumulative deaths

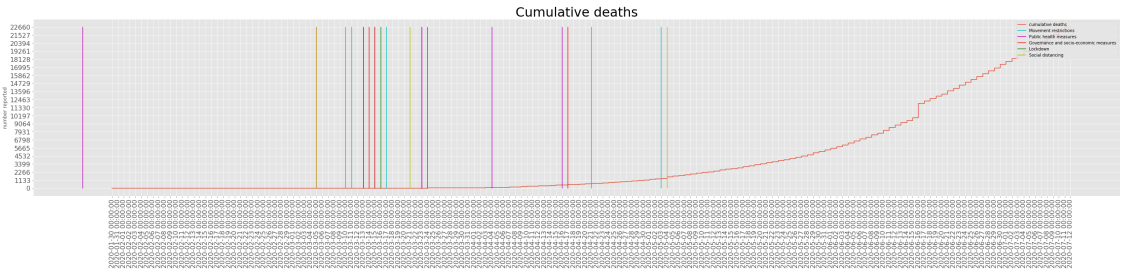
==========

India


New cases

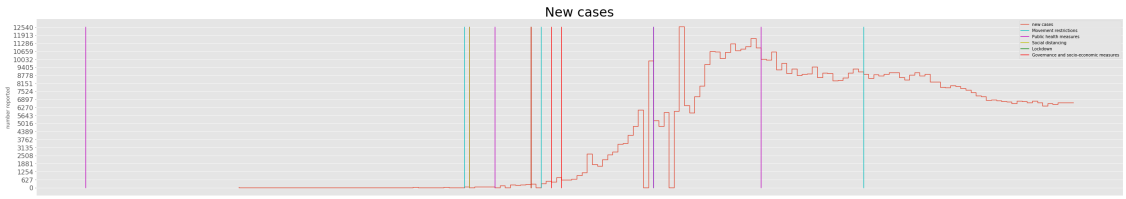

Cumulative cases

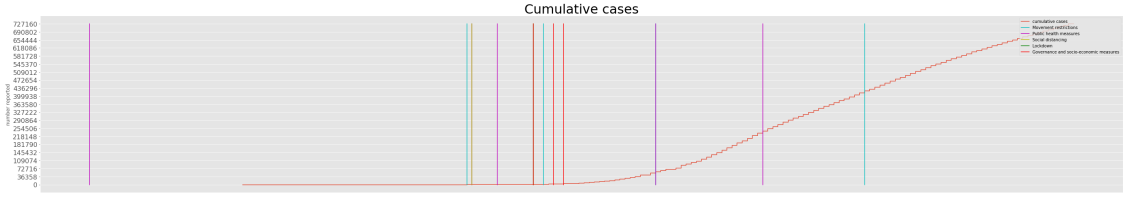

New deaths

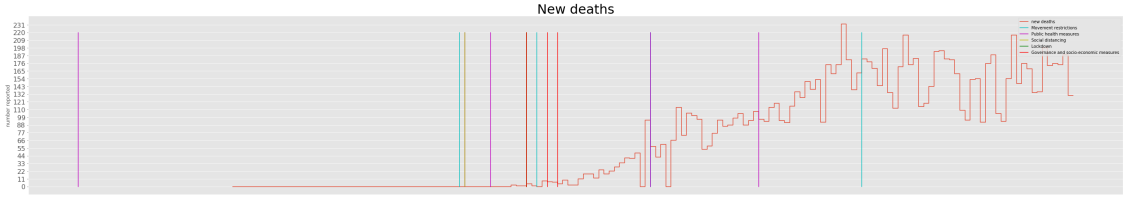

Cumulative deaths

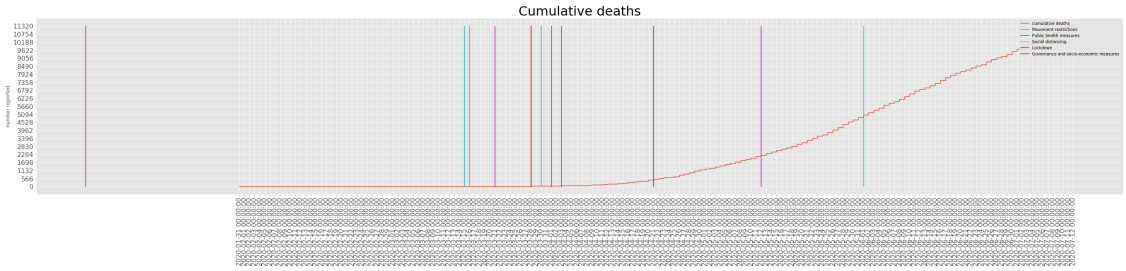
==========

Russia



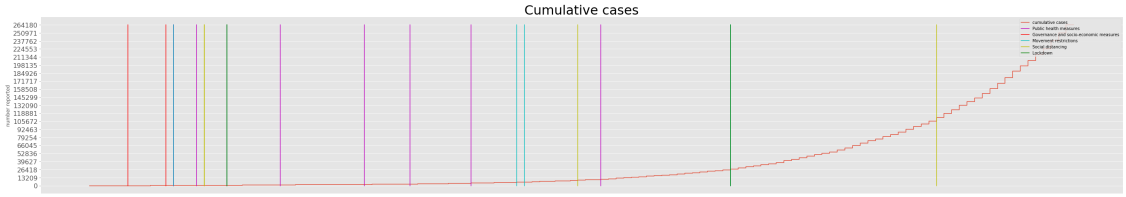


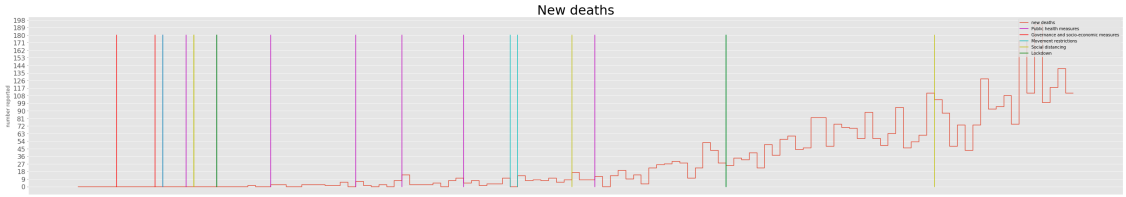


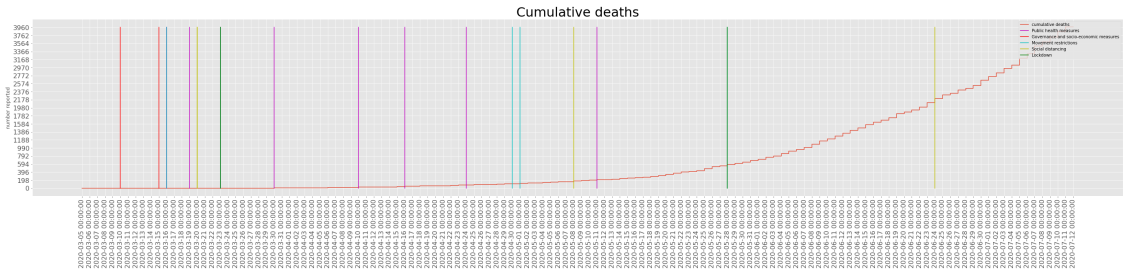


==========

South Africa







==========

5.0.5 Analysis of the effectiveness of government measures

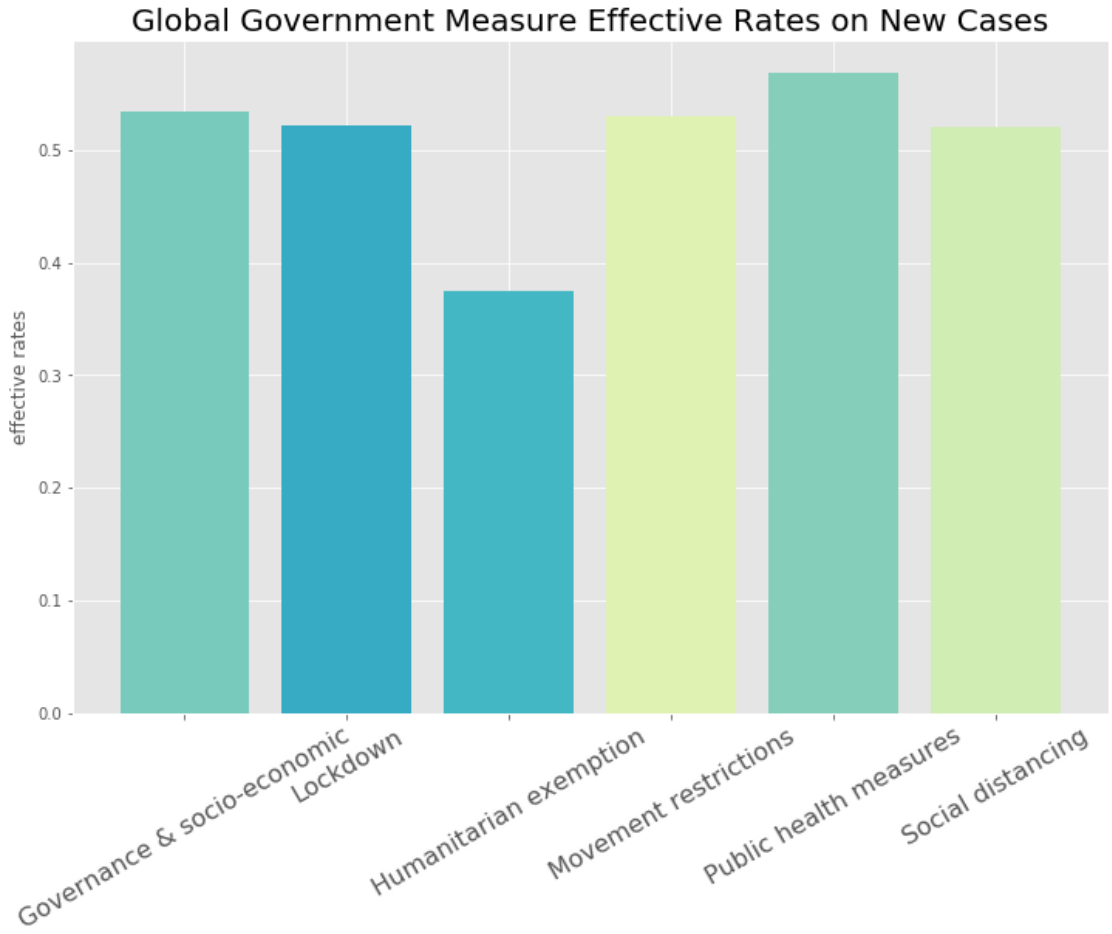
[191]:
```
def gov_correlation(shift_v,show_pic=False,print_res=False):
    cor_result={}
    cor_min={}
    cor_ratio={}
    series_tag=['New_cases','New_deaths']
```

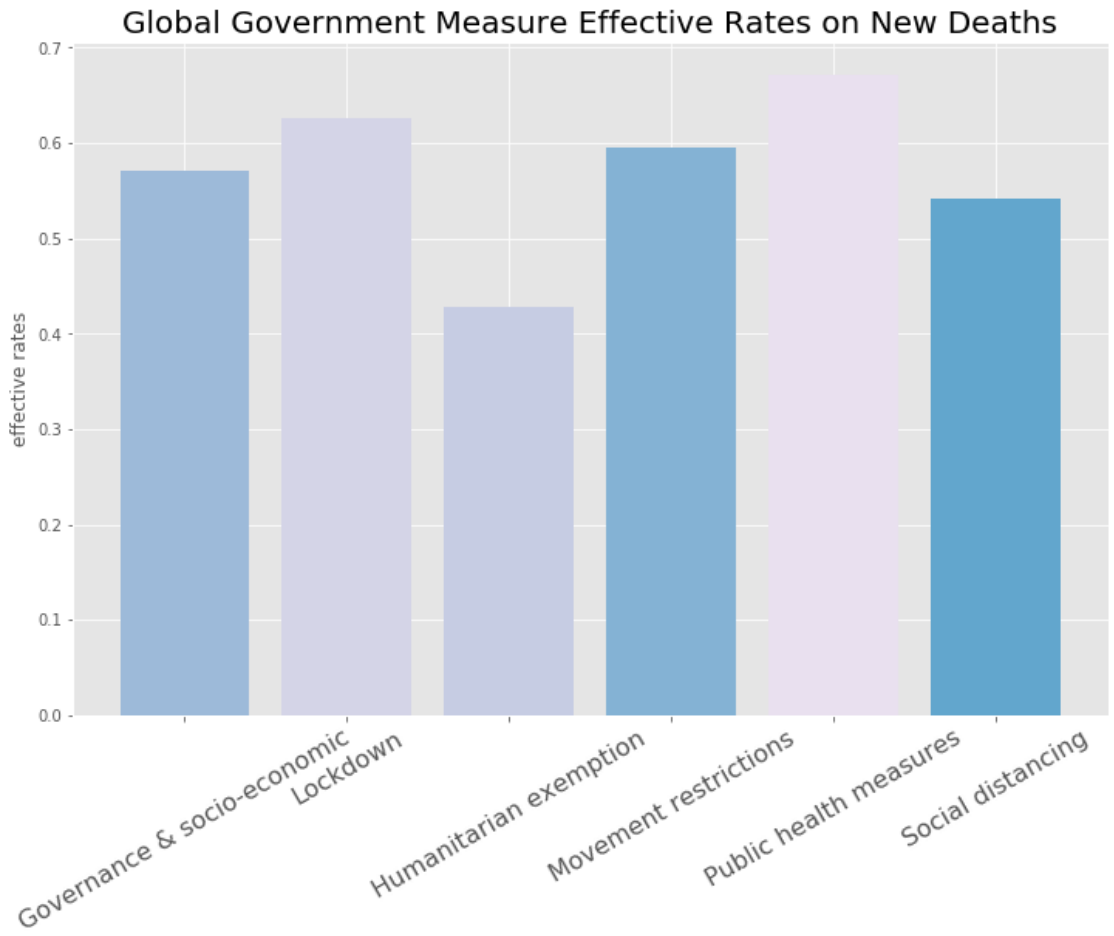
```
   measure_tag=["Governance and socio-economic measures",␣
↪"Lockdown","Humanitarian exemption","Movement restrictions","Public health␣
↪measures","Social distancing"]
   keyset=list(gov_by_measure.keys())
   keyset[keyset.index('Governance and socio-economic measures')] = 'Governance␣
↪& socio-economic'
   for series in series_tag:
       cor_result[series]={}
       cor_min[series]={}
       cor_ratio[series]=[]
   for series in series_tag:
       for tag in measure_tag:
           cor_result[series][tag]=[]
           cor_min[series][tag]=(1,"")
   for country in who_dict.keys():
       if country.strip() == "" or (iso_dict[country][0] not in gov_by_country.
↪keys()):
           continue
       date= who_dict[country]['Date_reported']

       ind = np.arange(len(who_dict[country]['Date_reported']))
       for i in range(len(series_tag)):
           series=series_tag[i]
           data=who_dict[country][series]
           delta=[data[0]]
           for i in range(1,len(data)):
               delta.append(data[i]-data[i-1])
           delta=np.array(delta)
           for key in gov_by_country[iso_dict[country][0]]:
               x=[0 for j in range(len(who_dict[country]['Date_reported']))]
               count=0
               # form the time series of the current number of measures in a␣
↪categories on day i for all days
               for sub in gov_by_country[iso_dict[country][0]][key].keys():
                   ␣
↪shift=min((gov_by_country[iso_dict[country][0]][key][sub][0]-date[0]).
↪days+shift_v,len(x)-1)
                   shift=max(shift,0)
                   count+=1
                   for d in range(shift,len(x)):
                       x[d]=count
               x=np.array(x)
               res=stats.pearsonr(x, delta)
               if key in measure_tag:
                   if not math.isnan(res[0]):
                       if res[0]<cor_min[series][key][0]:
                           cor_min[series][key]=(res[0],country)
```

```
                        cor_result[series][key].append(res[0])
    for series in series_tag:
        for tag in measure_tag:
            pos=np.sum(np.array(cor_result[series][tag]) > 0, axis=0)
            neg=np.sum(np.array(cor_result[series][tag]) < 0, axis=0)
            cor_ratio[series].append(neg/(pos+neg))
            if print_res:
                print(tag+"'s affects on "+' '.join(series.split('_')).lower())
                print("positive correlation countries: "+str(pos))
                print("negative correlation countries: "+str(neg))
                if cor_min[series][tag][1]!="":
                    print("measure works best in:␣
↪"+str(iso_dict[cor_min[series][tag][1]][1]))
                else:
                    print("no best countries found")
                print("")
    if show_pic==True:
        cm_dict={'New_cases':'YlGnBu','New_deaths':"PuBuGn"}
        for series in series_tag:
            plt.figure(figsize=(12,8))
            ind = np.arange(len(cor_ratio[series]))
            my_cmap = cm.get_cmap(cm_dict[series])
            color=[random.uniform(0.1, 0.6) for i in range(len(tags))]
            p1 = plt.bar(ind, cor_ratio[series],color=my_cmap(color))
            plt.ylabel('effective rates')
            plt.title('Global Government Measure Effective Rates on '+' '.
↪join(series.split('_')).title(),fontsize=20)
            plt.xticks(ind, keyset)
            plt.xticks(rotation=30,fontsize=16)
            plt.show()
    return cor_ratio
```

[192]:
```
gov_correlation(14,True)
```

```
C:\Users\nic_v\Anaconda3\lib\site-packages\scipy\stats\stats.py:3038:
RuntimeWarning: invalid value encountered in double_scalars
  r = r_num / r_den
```

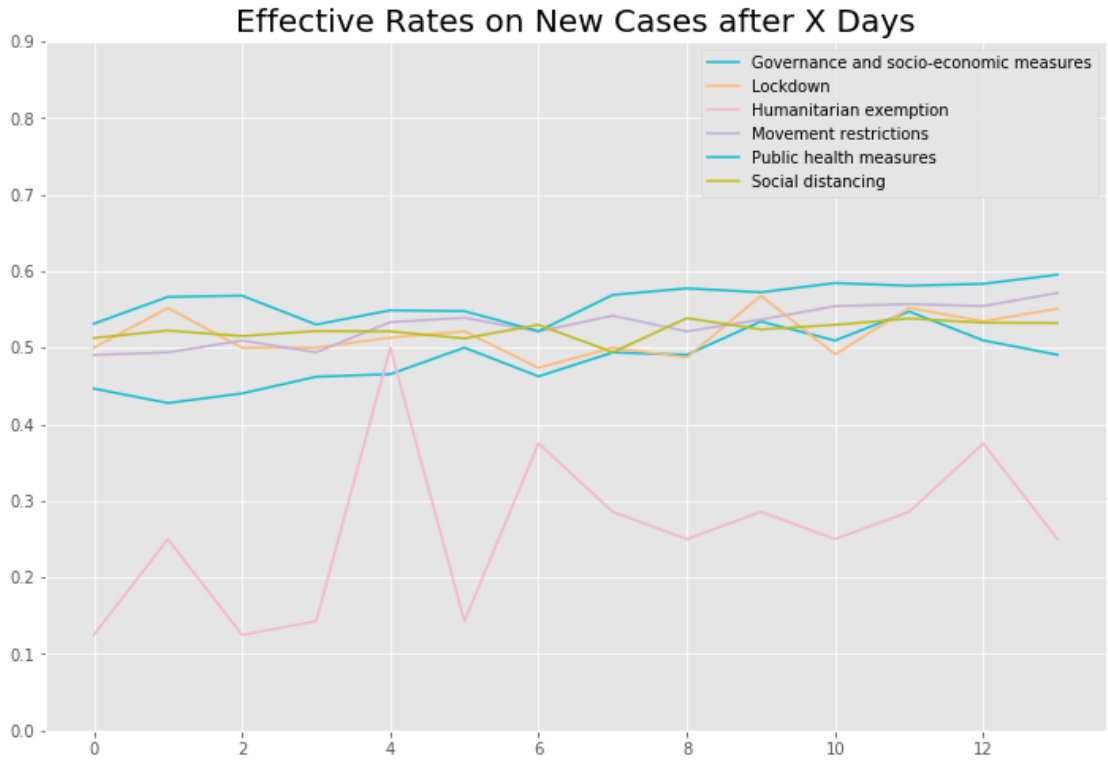

Global Government Measure Effective Rates on New Cases
effective rates
0.0
0.1
0.2
0.3
0.4
0.5
Governance & socio-economic
Lockdown
Humanitarian exemption
Movement restrictions
Public health measures
Social distancing

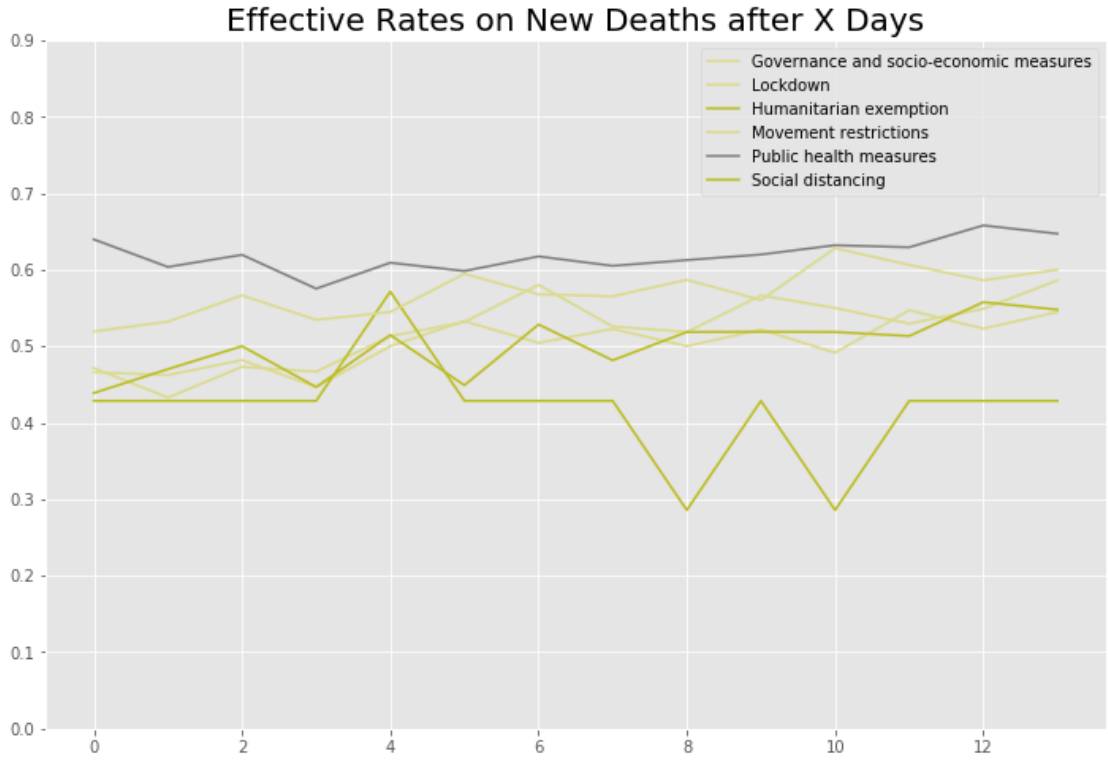


[192]:
```
{'New_cases': [0.5337423312883436,
  0.5210084033613446,
  0.375,
  0.5297619047619048,
  0.5688622754491018,
  0.5209580838323353],
 'New_deaths': [0.5702479338842975,
  0.6265060240963856,
  0.42857142857142855,
  0.5952380952380952,
  0.671875,
  0.5416666666666666]}
```

[193]:
```
cor_ratio={}
series_tag=['New_cases','New_deaths']
measure_tag=["Governance and socio-economic measures", "Lockdown","Humanitarian␣
 ↪exemption","Movement restrictions","Public health measures","Social␣
 ↪distancing"]
```

```
for series in series_tag:
    cor_ratio[series]={}
for series in series_tag:
    for tag in measure_tag:
        cor_ratio[series][tag]=[]

for i in range(14):
    dict_c=gov_correlation(i)
    for series in series_tag:
        for j in range(len(measure_tag)):
            tag=measure_tag[j]
            cor_ratio[series][tag].append(dict_c[series][j])

my_cmap = cm.get_cmap('tab20')
for series in series_tag:
    plt.figure(figsize=(12,8))
    for tag in measure_tag:
        ind = np.arange(len(cor_ratio[series][tag]))
        color=random.uniform(0, 1)
        plt.plot(ind, cor_ratio[series][tag],color=my_cmap(color),label=tag)
    plt.yticks(np.arange(0, 1, 0.1))
    plt.title("Effective Rates on "+' '.join(series.split('_')).title()+" after␣
 ↪X Days",fontsize=20)
    plt.legend(loc='upper right')
    plt.show()
```


Effective Rates on New Cases after X Days
Governance and socio-economic measures
Lockdown
Humanitarian exemption
Movement restrictions
Public health measures
Social distancing
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
2
4
6
8
10
12

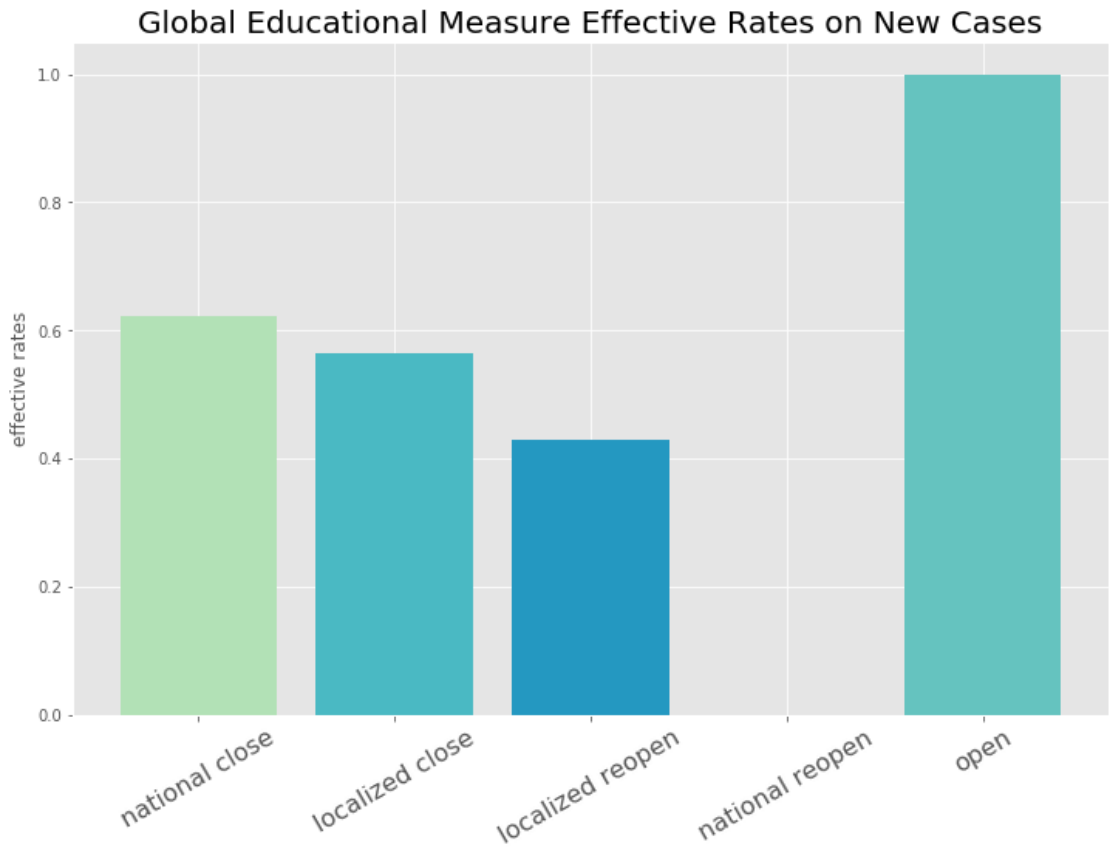

Effective Rates on New Deaths after X Days
Governance and socio-economic measures
Lockdown
Humanitarian exemption
Movement restrictions
Public health measures
Social distancing
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
2
4
6
8
10
12

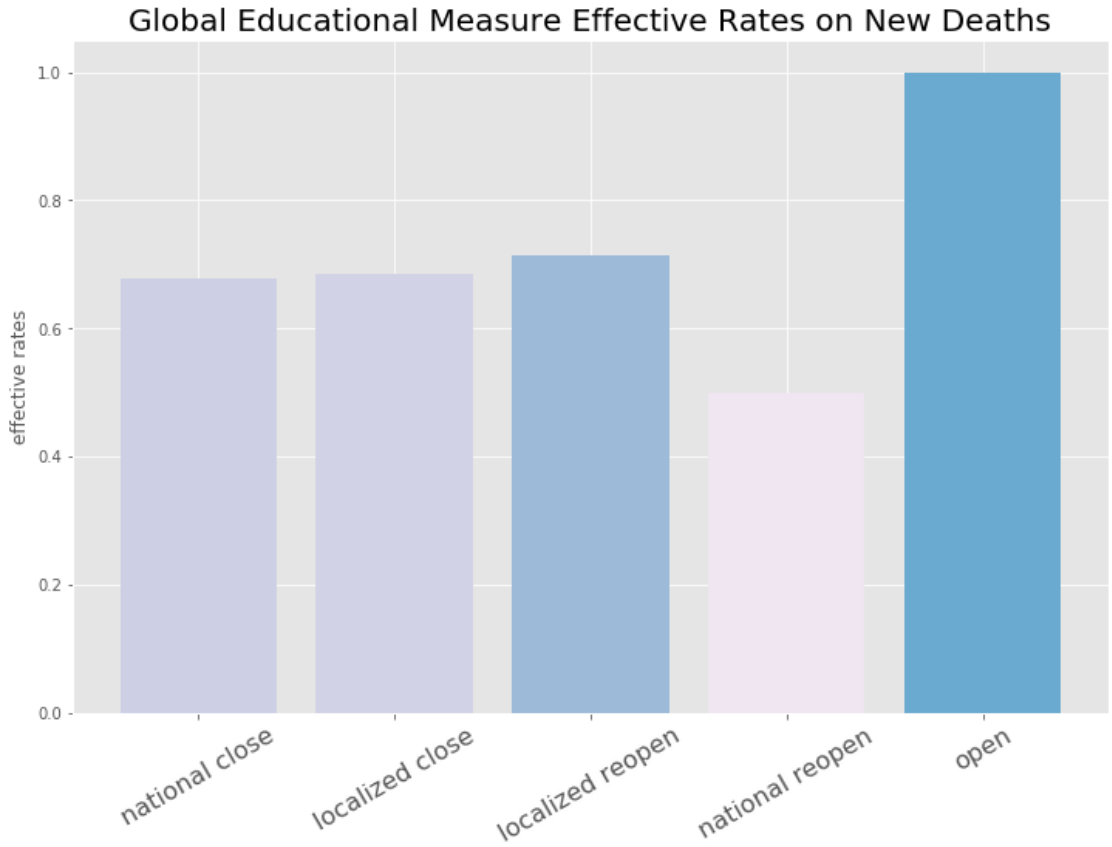
5.0.6 Analysis of the effectiveness of educational measures

[194]:
```
def edu_correlation(shift_v,show_pic=False,print_result=False):
    cor_result={}
    cor_min={}
    cor_ratio={}
    series_tag=['New_cases','New_deaths']
    measure_tag=["national close","localized close","localized reopen","national␣
↪reopen","open"]
    for series in series_tag:
        cor_result[series]={}
        cor_min[series]={}
        cor_ratio[series]=[]
    for series in series_tag:
        for tag in measure_tag:
            cor_result[series][tag]=[]
            cor_min[series][tag]=(1,"")


    for country in who_dict.keys():

        if country.strip() == "" or (iso_dict[country][0] not in edu_by_country.
↪keys()):
            continue
        date= who_dict[country]['Date_reported']

        ind = np.arange(len(who_dict[country]['Date_reported']))
        for i in range(len(series_tag)):
            series=series_tag[i]
            data=who_dict[country][series]
            delta=[data[0]]
            for i in range(1,len(data)):
                delta.append(data[i]-data[i-1])
            delta=np.array(delta)
            for key in edu_by_country[iso_dict[country][0]]:

                x=[False for j in range(len(who_dict[country]['Date_reported']))]
                shift=min((edu_by_country[iso_dict[country][0]][key]-date[0]).
↪days+shift_v,len(x)-1)
                shift=max(shift,0)
                x[shift]=True
                x=np.array(x)
                res=stats.pearsonr(x, delta)
                if key in measure_tag:
```

```
                    if not math.isnan(res[0]):
                        if res[0]<cor_min[series][key][0]:
                            cor_min[series][key]=(res[0],country)
                        cor_result[series][key].append(res[0])

    for series in series_tag:
        for tag in measure_tag:
            pos=np.sum(np.array(cor_result[series][tag]) > 0, axis=0)
            neg=np.sum(np.array(cor_result[series][tag]) < 0, axis=0)
            cor_ratio[series].append(neg/(pos+neg))
            if print_result:
                print("==========================================")
                print(tag+"'s affects on "+' '.join(series.split('_')).lower())
                print("positive correlation countries: "+str(pos))
                print("negative correlation countries: "+str(neg))
                print("measure works best in:␣
↪"+str(iso_dict[cor_min[series][tag][1]][1]))
                print("")

    if show_pic==True:
        for series in series_tag:
            cm_dict={'New_cases':'YlGnBu','New_deaths':"PuBuGn"}
            plt.figure(figsize=(12,8))
            ind = np.arange(len(cor_ratio[series]))
            color=[random.uniform(0.1, 0.6) for i in range(len(tags))]
            my_cmap = cm.get_cmap(cm_dict[series])
            p1 = plt.bar(ind, cor_ratio[series],color=my_cmap(color))
            plt.ylabel('effective rates')
            plt.title('Global Educational Measure Effective Rates on '+' '.
↪join(series.split('_')).title(),fontsize=20)
            plt.xticks(ind, measure_tag)
            plt.xticks(rotation=30,fontsize=16)
            plt.show()
    return cor_ratio
```

[195]:
```
edu_correlation(14,show_pic=True)
```

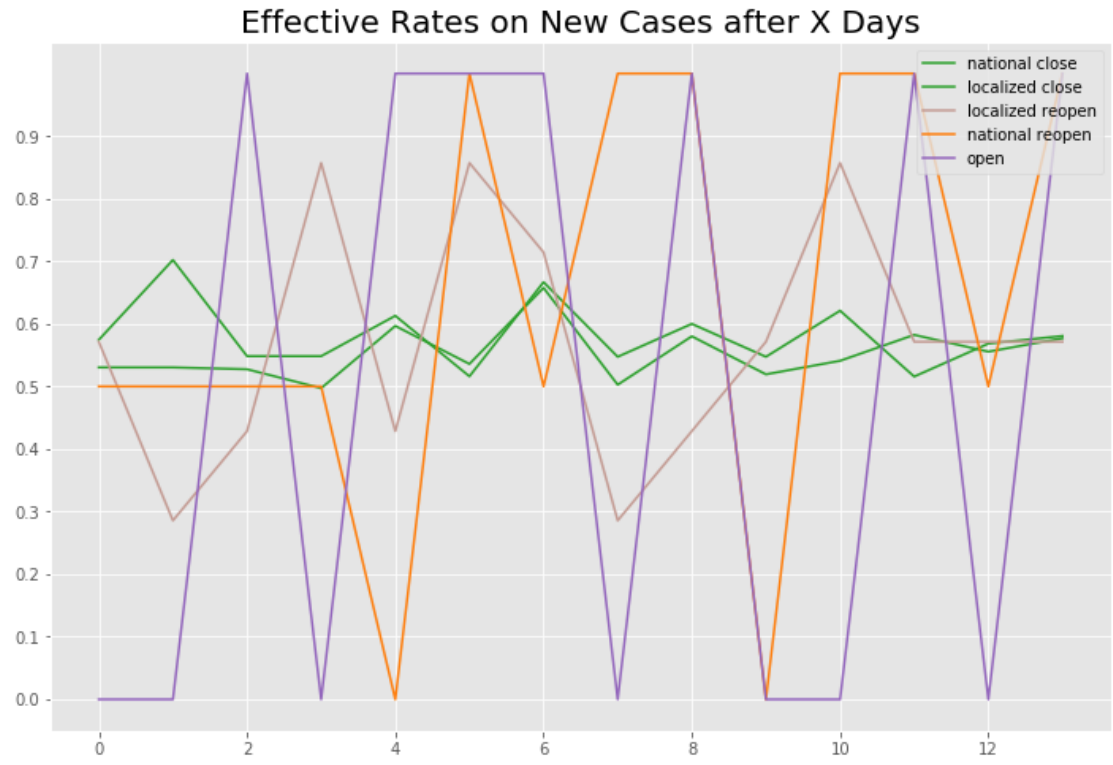

Global Educational Measure Effective Rates on New Cases
effective rates
1.0
0.8
0.6
0.4
0.2
0.0
national close
localized close
localized reopen
national reopen
open

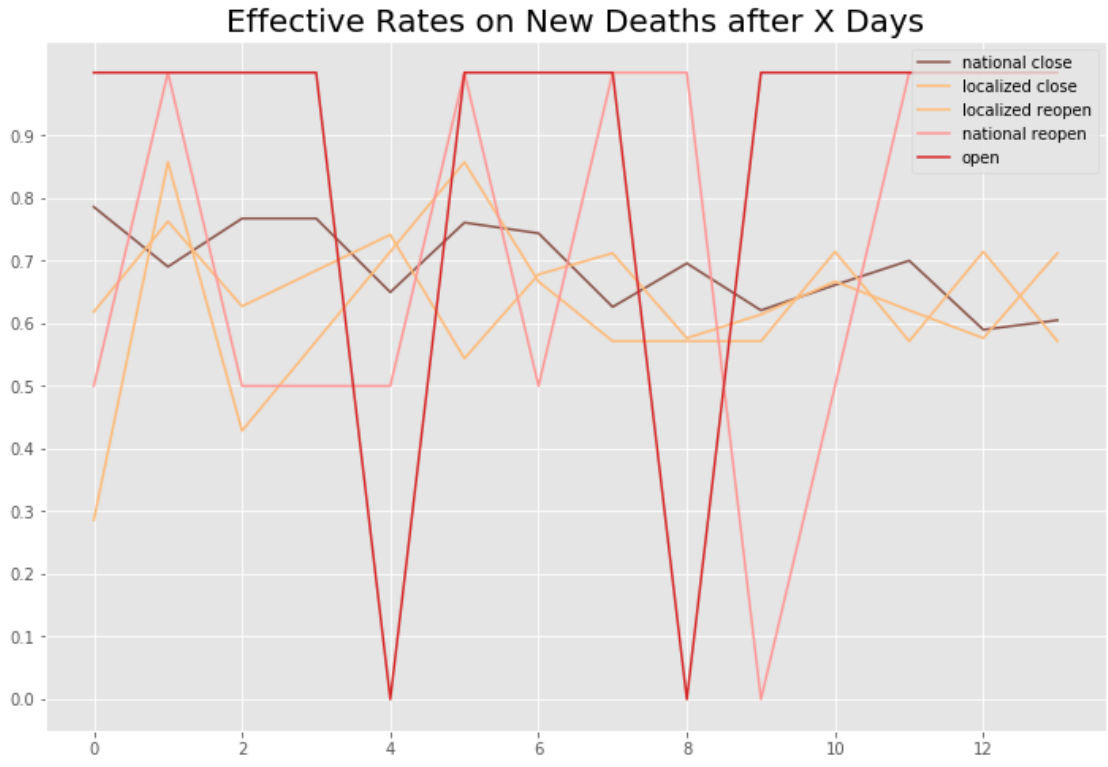


[195]:
```
{'New_cases': [0.6222222222222222,
  0.5638297872340425,
  0.42857142857142855,
  0.0,
  1.0],
 'New_deaths': [0.6774193548387096,
  0.6842105263157895,
  0.7142857142857143,
  0.5,
  1.0]}
```

[196]:
```
cor_ratio={}
series_tag=['New_cases','New_deaths']
measure_tag=["national close","localized close","localized reopen","national␣
 ↪reopen","open"]
for series in series_tag:
    cor_ratio[series]={}
for series in series_tag:
    for tag in measure_tag:
        cor_ratio[series][tag]=[]
```

```
for i in range(14):
    dict_c=edu_correlation(i)
    for series in series_tag:
        for j in range(len(measure_tag)):
            tag=measure_tag[j]
            cor_ratio[series][tag].append(dict_c[series][j])

my_cmap = cm.get_cmap('tab20')
for series in series_tag:
    plt.figure(figsize=(12,8))
    for tag in measure_tag:
        ind = np.arange(len(cor_ratio[series][tag]))
        color=random.uniform(0.1, 0.6)
        plt.plot(ind, cor_ratio[series][tag],color=my_cmap(color),label=tag)
    plt.yticks(np.arange(0, 1, 0.1))
    plt.title("Effective Rates on "+' '.join(series.split('_')).title()+" after␣
 ↪X Days",fontsize=20)
    plt.legend(loc='upper right')
    plt.show()
```




Effective Rates on New Deaths after X Days
national close
localized close
localized reopen
national reopen
open
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0
2
4
6
8
10
12