Proj3 Covid-19

June 9, 2020

[1]:
```
pip install chart-studio
```

```
Requirement already satisfied: chart-studio in
/srv/conda/envs/data100/lib/python3.7/site-packages (1.1.0)
Requirement already satisfied: plotly in
/srv/conda/envs/data100/lib/python3.7/site-packages (from chart-studio) (4.5.0)
Requirement already satisfied: six in
/srv/conda/envs/data100/lib/python3.7/site-packages (from chart-studio) (1.15.0)
Requirement already satisfied: requests in
/srv/conda/envs/data100/lib/python3.7/site-packages (from chart-studio) (2.22.0)
Requirement already satisfied: retrying>=1.3.3 in
/srv/conda/envs/data100/lib/python3.7/site-packages (from chart-studio) (1.3.3)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in
/srv/conda/envs/data100/lib/python3.7/site-packages (from requests->chart-
studio) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in
/srv/conda/envs/data100/lib/python3.7/site-packages (from requests->chart-
studio) (2020.4.5.1)
Requirement already satisfied: idna<2.9,>=2.5 in
/srv/conda/envs/data100/lib/python3.7/site-packages (from requests->chart-
studio) (2.8)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/srv/conda/envs/data100/lib/python3.7/site-packages (from requests->chart-
studio) (1.25.8)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
import chart_studio.plotly as py
import plotly.graph_objects as go #importing graphical objects
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
```

[3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

1 PROJECT 3: COVID-19

1.0.1 Data 100 Spring 2020 Final project

1.0.2 Sabrina Chiang, Susan Zhang, Makena Wilcox

1.1 Introduction

Through analyzing datasets of hospital resources against growing cases of coronavirus per county in the United State, we wanted to assess if hospitals have enough resources to treat the growing number COVID-19 patients. Specifically, we want to understand: what are the "current" hotspots of COVID-19 cases, do the amount of deaths in each county have an even ratio to the number of total cases in the same area, and does the amount of hospitals/ICU beds in each state impact the ratio of total cases to deaths in each state. We predict the percentage of deaths to cases will decrease with the number of hospitals and ICU beds available per county increases.

1.2 EDA and Data Cleaning

Filtering out unnecessary, imcomplete data, and filling in missing words from the 'abridged_couties.csv', 'covid19_confirmed_US.csv','time_series_covid19_deaths_US.csv' and data set. Removing columns except the last date of confirmed cases in the time series to get the most recent total number of cases from the time series dataset.

We named the abridged_counties.csv "features_data", the time_series_covid19_confirmed_US.csv "confirmed_data", and the time_series_covid19_deaths_US.csv "death_data". First we cleaned the features_data by removing feature columns that were not relevant to the questions we were trying to answer. We kept the columns that had the county FIPS as the primary key, the columns for county names, state names, and state abbreviations so that we could group our data based on counties and states, the latitude and longitude columns so that we could plot each county in our visualizations of the United States map, the column containing the population estimate of 2018 so that we could calculate what percent of each state got COVID-19, and the columns containing the number of hospitals and the number of ICU beds so that we could use these variables to train our linear regression model.

[4]:
```
features_data = pd.read_csv('abridged_couties.csv')
confirmed_data = pd.read_csv('covid19_confirmed_US.csv')
death_data = pd.read_csv('time_series_covid19_deaths_US.csv')
```

[5]:
```
features_data =␣
 ↪features_data[['countyFIPS','CountyName','StateName','State','PopulationEstimate2018','lat'
features_data
```

[5]:
```
   countyFIPS        CountyName StateName    State  PopulationEstimate2018  \
0       01001           Autauga        AL  Alabama                 55601.0
1       01003           Baldwin        AL  Alabama                218022.0
2       01005           Barbour        AL  Alabama                 24881.0
3       01007              Bibb        AL  Alabama                 22400.0
```

```
4         01009             Blount        AL  Alabama                 57840.0
…         …                 …       …     …                  …
3239      15005            Kalawao        HI       NaN                   88.0
3240      72039  Ciales Municipio        PR       NaN                15918.0
3241      72069  Humacao Municipio       PR       NaN                50532.0
3242      City1       New York City       NY       NaN                    NaN
3243      City2         Kansas City       MO       NaN                    NaN

            lat        lon  #Hospitals  #ICU_beds
0     32.540091 -86.645649         1.0        6.0
1     30.738314 -87.726272         3.0       51.0
2     31.874030 -85.397327         1.0        5.0
3     32.999024 -87.125260         1.0        0.0
4     33.990440 -86.562711         1.0        6.0
…           …         …          …         …
3239        NaN        NaN         0.0        0.0
3240        NaN        NaN         NaN        NaN
3241        NaN        NaN         NaN        NaN
3242        NaN        NaN         NaN        NaN
3243        NaN        NaN         NaN        NaN

[3244 rows x 9 columns]
```

Filling in missing data values for state column. We noticed that a lot of states were missing their corresponding state name.

[6]:
```
features_data['State'].iloc[67:94].fillna('Alaska', inplace=True)
features_data['State'].iloc[3234:3239].fillna('Alaska', inplace=True)
features_data['State'].iloc[2950:2953].fillna("Virginia", inplace=True)
features_data['State'].iloc[2912:2928].fillna("Virginia", inplace=True)
features_data['State'].iloc[2929:2938].fillna("Virginia", inplace=True)
features_data['State'].iloc[2938:2950].fillna("Virginia", inplace=True)
features_data['State'].iloc[2950:2953].fillna("Virginia", inplace=True)
features_data['State'].iloc[543:547].fillna("Hawaii", inplace=True)
features_data.loc[329,'State']= 'Florida'
features_data.loc[3239,'State']= 'Hawaii'
features_data.head()
```

[6]:
```
  countyFIPS CountyName StateName    State  PopulationEstimate2018        lat  \
0      01001    Autauga        AL  Alabama                 55601.0  32.540091
1      01003    Baldwin        AL  Alabama                218022.0  30.738314
2      01005    Barbour        AL  Alabama                 24881.0  31.874030
3      01007       Bibb        AL  Alabama                 22400.0  32.999024
4      01009     Blount        AL  Alabama                 57840.0  33.990440

         lon  #Hospitals  #ICU_beds
0 -86.645649         1.0        6.0
```

```
1 -87.726272         3.0        51.0
2 -85.397327         1.0         5.0
3 -87.125260         1.0         0.0
4 -86.562711         1.0         6.0
```

Wanted to see how many counties we had data for per state.

[7]:
```
state_counts = features_data['State'].value_counts().sort_values(ascending =␣
 ↪False)
state_counts.head()
```

[7]:
```
Texas       254
Georgia     159
Virginia    136
Kentucky    120
Missouri    115
Name: State, dtype: int64
```

Removing all the rows that do not include data from the 50 states in features data because this was not relevant to our analysis of the United States. We then dropped the last two rows because the format of the FIPS did not match the format of the rest of the other FIPS.

[8]:
```
features_data = features_data[features_data.StateName != 'PR'] #dropping PR␣
 ↪from table
features_data = features_data[features_data.StateName != 'MP'] #dropping MP␣
 ↪from table
features_data = features_data[features_data.StateName != 'GU'] #dropping GU␣
 ↪from table
features_data = features_data[features_data.StateName != 'AS'] #dropping AS␣
 ↪from table
features_data = features_data[features_data.StateName != 'VI'] #dropping VI␣
 ↪from table
features_data = features_data[:-2] #drops bottom two rows with no data
```

Cleaning the confirmed data and confirmed deaths to only include the rows FIPS, Admin2, Province State, Longitude, Latitute, and Cases since 4/18/20. We selected these columns so we can later use it to create visualizations.

[9]:
```
cleaned_data = confirmed_data[['FIPS', 'Admin2', 'Province_State','Lat',␣
 ↪'Long_','4/18/20']]
cleaned_data = cleaned_data.iloc[5:]
cleaned_data.rename(columns={'4/18/20':'Cases_4/18/20',
                             'Admin2':'CountyName',
                             'Province_State':'StateName',
                               'Long_':'Long'},
                  inplace=True)
cleaned_data = cleaned_data[:-3] #drops bottom three rows with no data;'Diamond␣
 ↪Princess',
```

```
#'District of Columbia','Grand Princess'
cleaned_data = cleaned_data[cleaned_data['StateName'] != 'Diamond Princess']
cleaned_data = cleaned_data[cleaned_data['StateName'] != 'District of Columbia']
cleaned_data.head()
```

[9]:

	FIPS	CountyName	StateName	Lat	Long	Cases_4/18/20
5	1001.0	Autauga	Alabama	32.539527	-86.644082	25
6	1003.0	Baldwin	Alabama	30.727750	-87.722071	109
7	1005.0	Barbour	Alabama	31.868263	-85.387129	18
8	1007.0	Bibb	Alabama	32.996421	-87.125115	26
9	1009.0	Blount	Alabama	33.982109	-86.567906	20

[10]:
```
cleaned_deaths = death_data[['FIPS', 'Admin2', 'Province_State','Lat',␣
↪'Long_','4/18/20']]
cleaned_deaths = cleaned_deaths.iloc[5:]
cleaned_deaths.rename(columns={'4/18/20':'Deaths_4/18/20',
                          'Admin2':'CountyName',
                          'Province_State':'StateName',
                            'Long_':'Long'},
                 inplace=True)
cleaned_deaths = cleaned_deaths[:-3] #drops bottom three rows with no data;
↪'Diamond Princess',
#'District of Columbia','Grand Princess'
cleaned_deaths = cleaned_deaths[cleaned_deaths['StateName'] != 'Diamond␣
↪Princess']
cleaned_deaths = cleaned_deaths[cleaned_deaths['StateName'] != 'District of␣
↪Columbia']
cleaned_deaths.head()
```

[10]:

	FIPS	CountyName	StateName	Lat	Long	Deaths_4/18/20
5	1001.0	Autauga	Alabama	32.539527	-86.644082	2
6	1003.0	Baldwin	Alabama	30.727750	-87.722071	2
7	1005.0	Barbour	Alabama	31.868263	-85.387129	0
8	1007.0	Bibb	Alabama	32.996421	-87.125115	0
9	1009.0	Blount	Alabama	33.982109	-86.567906	0

Found the total number of cases and deaths of COVID-19 per state after grouping by state name and then added the abbreviations of the state table. Labeled the tables total_per_state and total_deaths_state.

[11]:
```
total_per_state = cleaned_data.groupby('StateName')[['Cases_4/18/20']].sum()
```

[12]:
```
Abrevs = features_data.sort_values(by='State', ascending=False).
↪groupby('State').first()
total_per_state['Abrev']= Abrevs['StateName']
```

[13]:
```
total_per_state.sort_values('Cases_4/18/20',ascending = False).head()
```

[13]:
```
                Cases_4/18/20 Abrev
StateName
New York               241712    NY
New Jersey              81420    NJ
Massachusetts           36372    MA
Pennsylvania            31652    PA
California              30491    CA
```

[14]:
```
total_deaths_state = cleaned_deaths.groupby('StateName')[['Deaths_4/18/20']].
 ↪sum()
```

[15]:
```
Abrevs = features_data.sort_values(by='State', ascending=False).
 ↪groupby('State').first()
Abrevs
total_per_state['Abrev']= Abrevs['StateName']
total_deaths_state.sort_values('Deaths_4/18/20',ascending = False).head()
```

[15]:
```
                Deaths_4/18/20
StateName
New York                 17671
New Jersey                4070
Michigan                  2291
Massachusetts             1404
Louisiana                 1267
```

1.3 Visualizations

Made a choropleth that shows the number of coronavirus cases per state with color gradient legend to reflect the states with the most/least cases.

[16]:
```
data = dict(type='choropleth',
            locations = total_per_state['Abrev'],
            locationmode = 'USA-states',
            colorscale = 'Reds',
            text = total_per_state['Abrev'],
            z = total_per_state['Cases_4/18/20'],
            colorbar = {'title':"COVID-19 Cases"}
            )
layout = dict(title = 'COVID-19 Cases in USA Map',
              geo = dict(scope='usa')
             )
choromap = go.Figure(data = [data],layout = layout)
iplot(choromap)
```

1.4 Scattergeo Plot of COVID-19 Deaths and Cases

Making a scattergeo plot of the county's in the United States to see the clustering and number of COVID-19 cases/deaths. This visualization provides a color classification for each county depending on how confirmed cases are documented. From this we can recognize New York City has the highest number of cases compared to the other documented counties from the dot being colored red. One city in Southern California and a few others in the East Coast have a light blue color indicating they also have a higher number of cases. It is important to note that the amount of dots in each state are determined by the data provided in the CSV files, they do not show a density of cases.

[17]:
```
data = dict(
        type = 'scattergeo',
        locationmode = 'USA-states',
        mode = 'markers'
        )

data_high = data.copy()
data_high['lon'] = cleaned_data[cleaned_data['Cases_4/18/20'] > 250] ['Long']
data_high['lat'] = cleaned_data[cleaned_data['Cases_4/18/20'] > 250] ['Lat']
data_high['marker'] = dict(color = 'red', size=3)
data_high['name'] = '> 250 Cases'

data_med = data.copy()
data_med['lon'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 250] ['Long']
data_med['lat'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 250] ['Lat']
data_med['marker'] = dict(color = 'orange', size=3)
data_med['name'] = '< 250 Cases'

data_lowMed = data.copy()
data_lowMed['lon'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 150] ['Long']
data_lowMed['lat'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 150] ['Lat']
data_lowMed['marker'] = dict(color = 'lawngreen', size=3)
data_lowMed['name'] = '< 150 Cases'

data_low = data.copy()
data_low['lon'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 50] ['Long']
data_low['lat'] = cleaned_data[cleaned_data['Cases_4/18/20'] < 50] ['Lat']
data_low['marker'] = dict(color = 'blue', size=3)
data_low['name'] = '< 50 Cases'

layout = dict(
    title = 'COVID-19 County Cases in USA Map',
    geo = dict(
        scope = 'usa',
        projection = dict(type='albers usa'),
            ),
        )
```

```
fig = dict(data=[data_high, data_med, data_lowMed, data_low], layout=layout)
#plotly.offline.plot(fig)
iplot(fig)
```

[18]:
```
data = dict(
        type = 'scattergeo',
        locationmode = 'USA-states',
        mode = 'markers'
        )

data_high = data.copy()
data_high['lon'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] > 20]␣
 ↪['Long']
data_high['lat'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] > 20] ['Lat']
data_high['marker'] = dict(color = 'red', size=3)
data_high['name'] = '> 20 Deaths'

data_med = data.copy()
data_med['lon'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 20] ['Long']
data_med['lat'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 20] ['Lat']
data_med['marker'] = dict(color = 'orange', size=3)
data_med['name'] = '< 20 Deaths'

data_lowMed = data.copy()
data_lowMed['lon'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 10]␣
 ↪['Long']
data_lowMed['lat'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 10]␣
 ↪['Lat']
data_lowMed['marker'] = dict(color = 'lawngreen', size=3)
data_lowMed['name'] = '< 10 Deaths'

data_low = data.copy()
data_low['lon'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 5] ['Long']
data_low['lat'] = cleaned_deaths[cleaned_deaths['Deaths_4/18/20'] < 5] ['Lat']
data_low['marker'] = dict(color = 'blue', size=3)
data_low['name'] = '< 5 Deaths'

layout = dict(
    title = 'COVID-19 Deaths Per County in USA Map',
    geo = dict(
        scope = 'usa',
        projection = dict(type='albers usa'),
            ),
        )

fig = dict(data=[data_high, data_med, data_lowMed, data_low], layout=layout)
```

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

1.5 Merging Tables

Changed countyFIPS in features data to match cleaned data, so tables can be merged together. Also removed rows with certain FIPS numbers, as no data were present for each of those columns.

[19]:
```
features_data['countyFIPS'] = features_data['countyFIPS'].fillna(0.0).
↪astype(int)
```

[20]:
```
features_data = features_data[features_data.countyFIPS != 2201]
features_data = features_data[features_data.countyFIPS != 2232]
features_data = features_data[features_data.countyFIPS != 2280]
features_data = features_data[features_data.countyFIPS != 12025]
features_data = features_data[features_data.countyFIPS != 30113]
features_data = features_data[features_data.countyFIPS != 51560]
features_data = features_data[features_data.countyFIPS != 51780]
#FIPS: 2201, 2232, 2280, 12025,30113, 51560, 51780
#Remove these because there is no data in rows with this FIPS number
```

[21]:
```
cleaned_data['FIPS'] = cleaned_data['FIPS'].fillna(0.0).astype(int)
chopped_confirmed = cleaned_data[['FIPS', 'Cases_4/18/20']]
```

We merged the three datasets on the primary key, FIPS. We only keep the columns, 'FIPS' and '4/18/20', in both confirmed_data and death_data so that we know how many confirmed cases and deaths there are for each FIPS in features_data. Confirmed_data and death_data have more rows than features_data because we did not clean them, but we did not have to clean them because when we merged on FIPS, the merged data frame only kept the rows that we needed. This new merged data frame includes all the features, the number of cases, and the number of deaths which we need to use for our cross-validation training.

[22]:
```
#merging the datasets on FIPS
merged_cases = pd.merge(features_data, chopped_confirmed, left_on='countyFIPS',␣
↪right_on='FIPS')
#drop the FIPS column because it is the same as countyFIPS
merged_cases = merged_cases.drop(['FIPS'], axis=1)
merged_cases
```

[22]:
```
      countyFIPS                          CountyName StateName    State  \
0           1001                             Autauga        AL  Alabama
1           1003                             Baldwin        AL  Alabama
2           1005                             Barbour        AL  Alabama
3           1007                                Bibb        AL  Alabama
4           1009                              Blount        AL  Alabama
…            …                                   …         …        …
```

```
3134         2195                   Petersburg Borough        AK   Alaska
3135         2198  Prince of Wales-Hyder Census Area        AK   Alaska
3136         2230                 Skagway Municipality        AK   Alaska
3137         2275            Wrangell City and Borough        AK   Alaska
3138        15005                              Kalawao        HI   Hawaii

      PopulationEstimate2018        lat        lon  #Hospitals  #ICU_beds  \
0                    55601.0  32.540091 -86.645649         1.0        6.0
1                   218022.0  30.738314 -87.726272         3.0       51.0
2                    24881.0  31.874030 -85.397327         1.0        5.0
3                    22400.0  32.999024 -87.125260         1.0        0.0
4                    57840.0  33.990440 -86.562711         1.0        6.0
…                          …          …          …           …          …
3134                  3221.0        NaN        NaN         1.0        0.0
3135                  6422.0        NaN        NaN         0.0        0.0
3136                  1148.0        NaN        NaN         0.0        0.0
3137                  2503.0        NaN        NaN         1.0        0.0
3138                    88.0        NaN        NaN         0.0        0.0

      Cases_4/18/20
0                25
1               109
2                18
3                26
4                20
…               …
3134              2
3135              2
3136              0
3137              0
3138              0

[3139 rows x 10 columns]
```

[23]:
```
merged_deaths = pd.merge(merged_cases, cleaned_deaths, left_on='countyFIPS',␣
 ↪right_on='FIPS')
merged_deaths = merged_deaths.
 ↪drop(['FIPS','Lat','Long','CountyName_y','StateName_y'], axis=1)
merged_deaths
```

[23]:
```
      countyFIPS                      CountyName_x StateName_x    State  \
0           1001                           Autauga          AL  Alabama
1           1003                           Baldwin          AL  Alabama
2           1005                           Barbour          AL  Alabama
3           1007                              Bibb          AL  Alabama
4           1009                            Blount          AL  Alabama
…            …                                 …          …        …
```

```
3134        2195                   Petersburg Borough          AK    Alaska
3135        2198  Prince of Wales-Hyder Census Area          AK    Alaska
3136        2230                 Skagway Municipality          AK    Alaska
3137        2275            Wrangell City and Borough          AK    Alaska
3138       15005                              Kalawao          HI    Hawaii

      PopulationEstimate2018        lat        lon  #Hospitals  #ICU_beds  \
0                    55601.0  32.540091 -86.645649         1.0        6.0
1                   218022.0  30.738314 -87.726272         3.0       51.0
2                    24881.0  31.874030 -85.397327         1.0        5.0
3                    22400.0  32.999024 -87.125260         1.0        0.0
4                    57840.0  33.990440 -86.562711         1.0        6.0
…                        …        …         …          …         …
3134                  3221.0        NaN        NaN         1.0        0.0
3135                  6422.0        NaN        NaN         0.0        0.0
3136                  1148.0        NaN        NaN         0.0        0.0
3137                  2503.0        NaN        NaN         1.0        0.0
3138                    88.0        NaN        NaN         0.0        0.0

      Cases_4/18/20  Deaths_4/18/20
0                25               2
1               109               2
2                18               0
3                26               0
4                20               0
…               …              …
3134              2               0
3135              2               0
3136              0               0
3137              0               0
3138              0               0

[3139 rows x 11 columns]
```

Grouped the merged table by State and StateName_X, while sorting the table by the %Cases/Population.

[24]:
```
merged_per_state = merged_deaths.
 ↪groupby(['State','StateName_x'])[['PopulationEstimate2018','Deaths_4/18/
 ↪20','Cases_4/18/20','#Hospitals','#ICU_beds']].sum()
merged_per_state['%Deaths/Cases'] = (merged_per_state['Deaths_4/18/20']/
 ↪merged_per_state['Cases_4/18/20']) * 100
merged_per_state['%Cases/Population'] = (merged_per_state['Cases_4/18/20']/
 ↪merged_per_state['PopulationEstimate2018']) * 100
merged_per_state.sort_values('%Cases/Population',ascending = False)
```

[24]:

```
                              PopulationEstimate2018  Deaths_4/18/20  \
State          StateName_x
New York       NY                         19542209.0           16612
New Jersey     NJ                          8908520.0            4068
Massachusetts  MA                          6902149.0            1384
Louisiana      LA                          4659978.0            1266
Connecticut    CT                          3572665.0            1083
Rhode Island   RI                          1057315.0               3
Michigan       MI                          9995915.0            2286
Delaware       DE                           967171.0              67
Pennsylvania   PA                         12807060.0            1042
Illinois       IL                         12741080.0            1256
Maryland       MD                          6042718.0             421
South Dakota   SD                           867926.0               7
Indiana        IN                          6691878.0             545
Georgia        GA                         10519475.0             666
Colorado       CO                          5695564.0             388
Washington     WA                          7535591.0             613
Mississippi    MS                          2986530.0             152
Vermont        VT                           626299.0              37
Florida        FL                         21299325.0             748
Nevada         NV                          3034392.0             151
New Hampshire  NH                          1356458.0               3
Alabama        AL                          4887871.0             153
Virginia       VA                          8517685.0             164
Idaho          ID                          1754208.0              43
Tennessee      TN                          6770010.0             141
Utah           UT                          3161105.0              25
Ohio           OH                         11689442.0             451
New Mexico     NM                          2095428.0              53
Missouri       MO                          6126452.0             184
South Carolina SC                          5084127.0             119
Iowa           IA                          3156145.0              74
California     CA                         39557045.0            1140
Wisconsin      WI                          5813568.0             212
North Dakota   ND                           760077.0               9
Arizona        AZ                          7171646.0             180
Texas          TX                         28701845.0             476
Maine          ME                          1338404.0              32
Kansas         KS                          2911505.0              85
Nebraska       NE                          1929268.0              15
North Carolina NC                         10383620.0             187
Oklahoma       OK                          3943079.0             131
Kentucky       KY                          4468402.0             138
Arkansas       AR                          3013825.0              38
Wyoming        WY                           577737.0               1
Oregon         OR                          4190713.0              72
```

West Virginia	WV	1805832.0	7
Alaska	AK	729135.0	5
Montana	MT	1062305.0	10
Hawaii	HI	1420491.0	9
Minnesota	MN	5611179.0	121

		Cases_4/18/20	#Hospitals	#ICU_beds \
State	StateName_x			
New York	NY	241712	165.0	3952.0
New Jersey	NJ	80672	64.0	1822.0
Massachusetts	MA	35616	58.0	1326.0
Louisiana	LA	23523	111.0	1289.0
Connecticut	CT	17025	30.0	674.0
Rhode Island	RI	3345	10.0	279.0
Michigan	MI	30074	130.0	2423.0
Delaware	DE	2508	6.0	186.0
Pennsylvania	PA	31652	162.0	3169.0
Illinois	IL	29076	176.0	3144.0
Maryland	MD	12326	47.0	1134.0
South Dakota	SD	1541	56.0	152.0
Indiana	IN	10641	118.0	1861.0
Georgia	GA	16634	129.0	2508.0
Colorado	CO	8980	80.0	1095.0
Washington	WA	11332	88.0	1265.0
Mississippi	MS	3974	95.0	824.0
Vermont	VT	793	14.0	94.0
Florida	FL	25489	178.0	5604.0
Nevada	NV	3592	36.0	900.0
New Hampshire	NH	1342	26.0	242.0
Alabama	AL	4712	86.0	1533.0
Virginia	VA	8053	81.0	1654.0
Idaho	ID	1655	42.0	314.0
Tennessee	TN	6293	100.0	2209.0
Utah	UT	2917	45.0	565.0
Ohio	OH	10222	156.0	3314.0
New Mexico	NM	1798	42.0	340.0
Missouri	MO	5167	106.0	1888.0
South Carolina	SC	4248	57.0	1225.0
Iowa	IA	2512	116.0	545.0
California	CA	30491	329.0	7338.0
Wisconsin	WI	4199	122.0	1159.0
North Dakota	ND	528	44.0	238.0
Arizona	AZ	4724	76.0	1559.0
Texas	TX	18704	384.0	6199.0
Maine	ME	846	33.0	256.0
Kansas	KS	1821	132.0	767.0
Nebraska	NE	1189	87.0	440.0

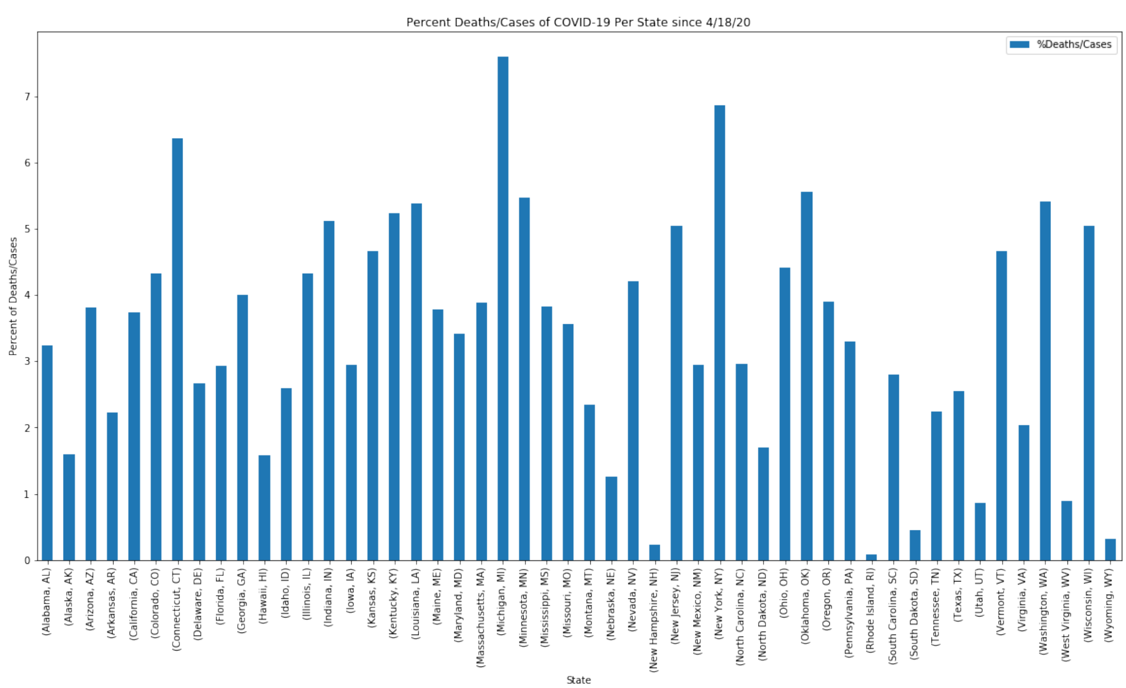
North Carolina	NC	6328	106.0	2227.0
Oklahoma	OK	2357	118.0	1064.0
Kentucky	KY	2634	91.0	1392.0
Arkansas	AR	1702	74.0	732.0
Wyoming	WY	309	27.0	102.0
Oregon	OR	1844	59.0	663.0
West Virginia	WV	785	49.0	653.0
Alaska	AK	314	22.0	119.0
Montana	MT	426	62.0	165.0
Hawaii	HI	568	21.0	201.0
Minnesota	MN	2209	127.0	1171.0

		%Deaths/Cases	%Cases/Population
State	StateName_x		
New York	NY	6.872642	1.236871
New Jersey	NJ	5.042642	0.905560
Massachusetts	MA	3.885894	0.516013
Louisiana	LA	5.381967	0.504788
Connecticut	CT	6.361233	0.476535
Rhode Island	RI	0.089686	0.316367
Michigan	MI	7.601250	0.300863
Delaware	DE	2.671451	0.259313
Pennsylvania	PA	3.292051	0.247145
Illinois	IL	4.319714	0.228207
Maryland	MD	3.415544	0.203981
South Dakota	SD	0.454250	0.177550
Indiana	IN	5.121699	0.159014
Georgia	GA	4.003848	0.158126
Colorado	CO	4.320713	0.157667
Washington	WA	5.409460	0.150380
Mississippi	MS	3.824862	0.133064
Vermont	VT	4.665826	0.126617
Florida	FL	2.934599	0.119670
Nevada	NV	4.203786	0.118376
New Hampshire	NH	0.223547	0.098934
Alabama	AL	3.247029	0.096402
Virginia	VA	2.036508	0.094544
Idaho	ID	2.598187	0.094345
Tennessee	TN	2.240585	0.092954
Utah	UT	0.857045	0.092278
Ohio	OH	4.412052	0.087446
New Mexico	NM	2.947720	0.085806
Missouri	MO	3.561061	0.084339
South Carolina	SC	2.801318	0.083554
Iowa	IA	2.945860	0.079591
California	CA	3.738808	0.077081
Wisconsin	WI	5.048821	0.072228

North Dakota	ND	1.704545	0.069467
Arizona	AZ	3.810330	0.065871
Texas	TX	2.544910	0.065167
Maine	ME	3.782506	0.063210
Kansas	KS	4.667765	0.062545
Nebraska	NE	1.261564	0.061630
North Carolina	NC	2.955120	0.060942
Oklahoma	OK	5.557913	0.059776
Kentucky	KY	5.239180	0.058947
Arkansas	AR	2.232667	0.056473
Wyoming	WY	0.323625	0.053485
Oregon	OR	3.904555	0.044002
West Virginia	WV	0.891720	0.043470
Alaska	AK	1.592357	0.043065
Montana	MT	2.347418	0.040101
Hawaii	HI	1.584507	0.039986
Minnesota	MN	5.477592	0.039368

Made a bar graph with all of the 50 different states to show the relative sizes of the %Deaths/Cases for each state.

[25]:
```
merged_per_state[['%Deaths/Cases']].plot(kind = 'bar',figsize = (20,10))
plt.xticks(rotation=90)
plt.xlabel('State')
plt.ylabel('Percent of Deaths/Cases')
plt.title('Percent Deaths/Cases of COVID-19 Per State since 4/18/20')
plt.show();
```



1.6 Cross-Validation and Linear Regression

We split our merged data set into 20% testing and 80% training sets and trained our linear model on two different features: number of ICUs and number of hospitals. Then, created a scatter plot of the actual death rates to the predicted death rates.

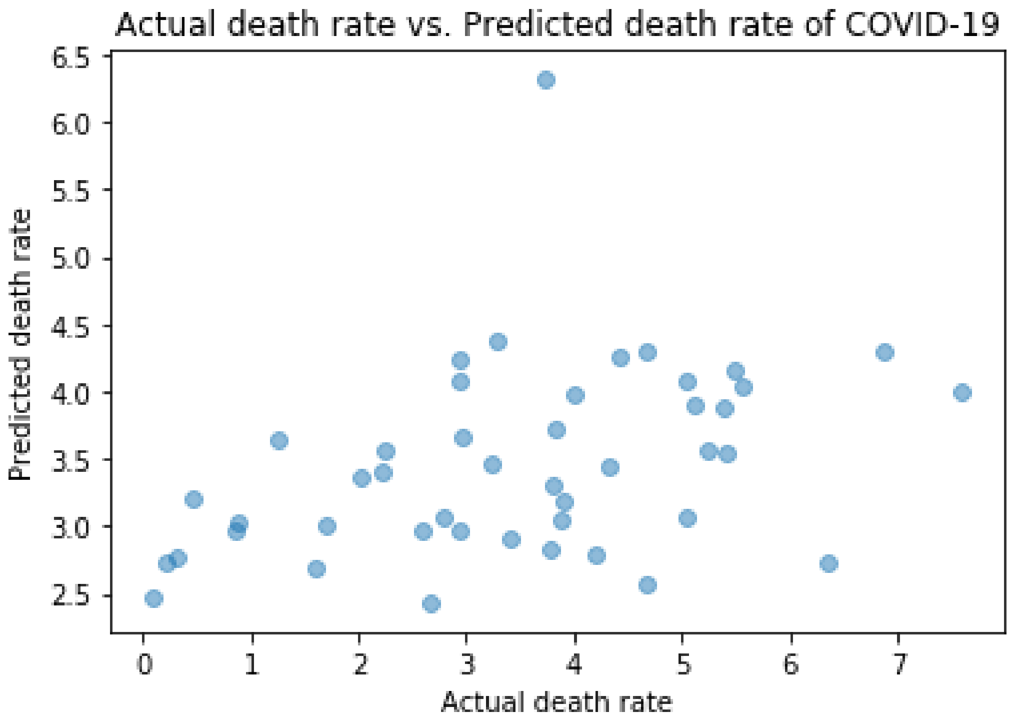
[26]:
```
from sklearn.model_selection import train_test_split
```

[27]:
```
tr, te = train_test_split(merged_per_state, test_size=0.1, random_state=83)
```

[28]:
```
def phi(df):
    return df[["#Hospitals", "#ICU_beds"]]
```

[29]:
```
X_train = phi(tr)
X_test = phi(te)
Y_train = tr['%Deaths/Cases']
Y_test = te['%Deaths/Cases']
```

[30]:
```
import sklearn.linear_model as lm
from sklearn.linear_model import LinearRegression

linear_model = lm.LinearRegression()

# Fit the linear model
linear_model.fit(X_train, Y_train)

# Predict percent of deaths per # of cases on the test set
Y_pred = linear_model.predict(X_train)

# Plot predicted vs true %Deaths/Cases
plt.scatter(Y_train, Y_pred, alpha=0.5)
plt.xlabel("Actual death rate")
plt.ylabel("Predicted death rate")
plt.title("Actual death rate vs. Predicted death rate of COVID-19");
```



After training our data, we calculated a training and testing error with our rsme function.

[31]:
```
def rmse(y, yhat):
    return np.sqrt(np.mean((y - yhat)**2))
```

[32]:
```
train_error = rmse(Y_train, linear_model.predict(X_train))
test_error = rmse(Y_test, linear_model.predict(X_test))

print("Training Error (RMSE):", train_error)
print("Testing Error (RMSE):", test_error)
```

```
Training Error (RMSE): 1.643019404049064
Testing Error (RMSE): 2.252984101653196
```

To further assess if we had made a good model, we used 5-fold cross validation. We normalized the data and found the hyperparameter with the smallest cross-validation error. We found that the best alpha value and the cross validation error for this alpha value.

[33]:
```
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
```

[34]:
```
def rmse_score(model, X, y):
    return np.sqrt(np.mean((y - model.predict(X)) ** 2))
```

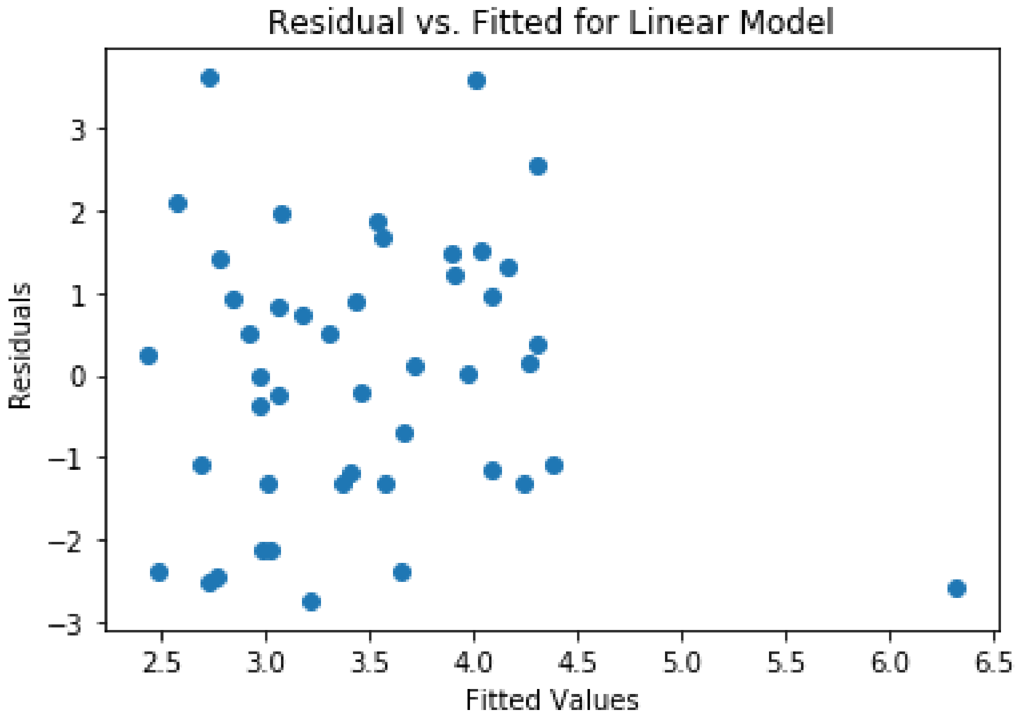
[35]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Ridge
from sklearn.model_selection import cross_val_score

alpha_arr = np.linspace(0.02, 0.5, 60)
cv_errors = []

model = Pipeline([
    ("transformer", StandardScaler()),
    ("LinearModel", Ridge(alpha=0.1))
])

for alpha in alpha_arr:
    model.set_params(LinearModel__alpha=alpha)

# compute the cross validation error
    cv_error = np.mean(cross_val_score(model, X_train, Y_train,␣
 ↪scoring=rmse_score, cv=5))

    cv_errors.append(cv_error)

min_cv_error = min(cv_errors)
index_of_min_cv_error = cv_errors.index(min_cv_error)
best_alpha_ridge = alpha_arr[index_of_min_cv_error]

print(f"The best alpha value is {best_alpha_ridge}")
print(f"Cross validation error for the best alpha value is {cv_errors[np.
 ↪argmin(cv_errors)]}")
```

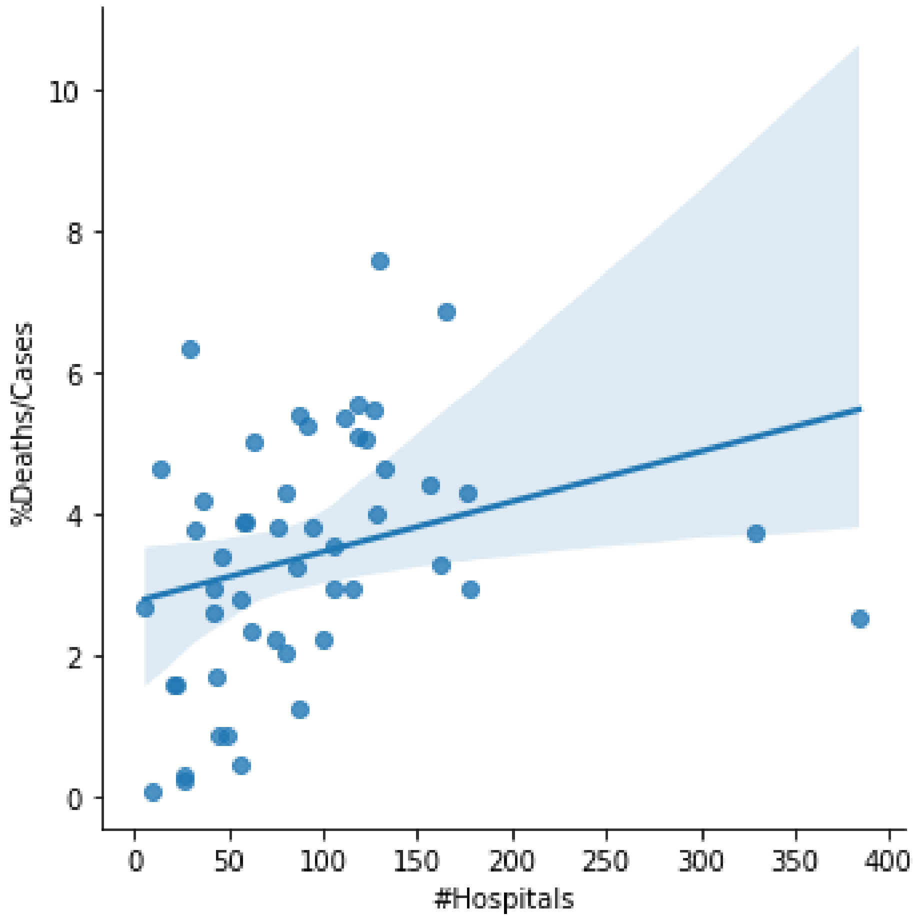
```
The best alpha value is 0.5
Cross validation error for the best alpha value is 1.9175874842651077
```

In order to visualize our errors and test to see if a linear regression model was good for the data, we created a residual plot. We then made a regression line to determine if there was correlation between the number of hospitals and the number of ICUs with COVID-19 death rate for each state.

[36]:
```
y_fitted = linear_model.predict(X_train)
plt.scatter(y_fitted, Y_train - y_fitted)
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.title('Residual vs. Fitted for Linear Model')
sns.lmplot(x='#Hospitals',y='%Deaths/Cases',data=merged_per_state,fit_reg=True)
```

[36]: <seaborn.axisgrid.FacetGrid at 0x7f29b4cbbf50>


Residual vs. Fitted for Linear Model
Residuals
Fitted Values
3
2
1
0
−1
−2
−3
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5



1.7 Congratulations!

You are finished with this assignment. Please don't forget to submit by 11:59pm PST on Wednesday, 05/13!

1.8 Submit

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. **Please save before submitting!**

[ ]:

```

```

[ ]:

```

```