Assignment #4

Step 1:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
zeroaccess_df = pd.read_csv("zeroaccess.csv")
state_df = pd.read_csv("state-internets.csv")
county_df = pd.read_csv("county-data.csv")
print(zeroaccess_df.shape,state_df.shape, county_df.shape)
```

```
(808446, 2) (49, 3) (3072, 6)
```

In [3]:
```
zeroaccess_df.head()
```

Out[3]:

	lat	long
0	-10.0000	-55.0000
1	38.0888	-78.5592
2	38.9990	-84.6266
3	48.6210	7.4944
4	43.2342	-86.2484

Step 2:

File Edit View Insert Cell Kernel Widgets Help

Run Code

4	43.2342	-86.2484

In [4]:
```
state_df.head()
```

Out[4]:

	state	population	internet
0	Alabama	4758191	3092273
1	Arizona	6665093	5230474
2	Arkansas	2919815	1949869
3	California	37350092	29758896
4	Colorado	5077553	4058749

In [5]:
```
county_df.head()
```

Out[5]:

	subregion	region	pop	income	ipaddr	ufo2010
0	abbeville	south carolina	25101	34670	30330	2
1	acadia	louisiana	61912	37970	38203	6
2	accomack	virginia	33341	41595	41338	2
3	ada	idaho	409061	55304	1035427	59
4	adair	iowa	7481	47623	3762	0

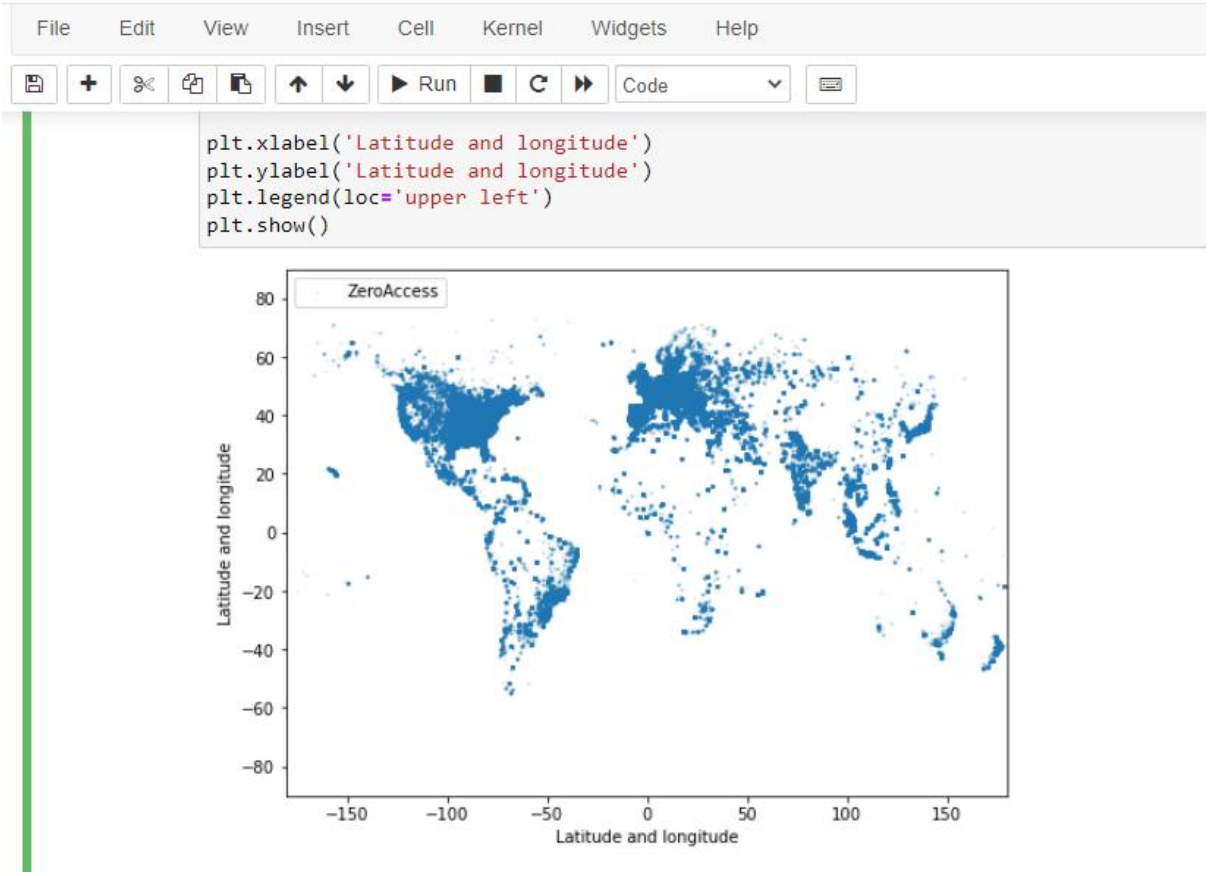
Step 3:

File Edit View Insert Cell Kernel Widgets Help

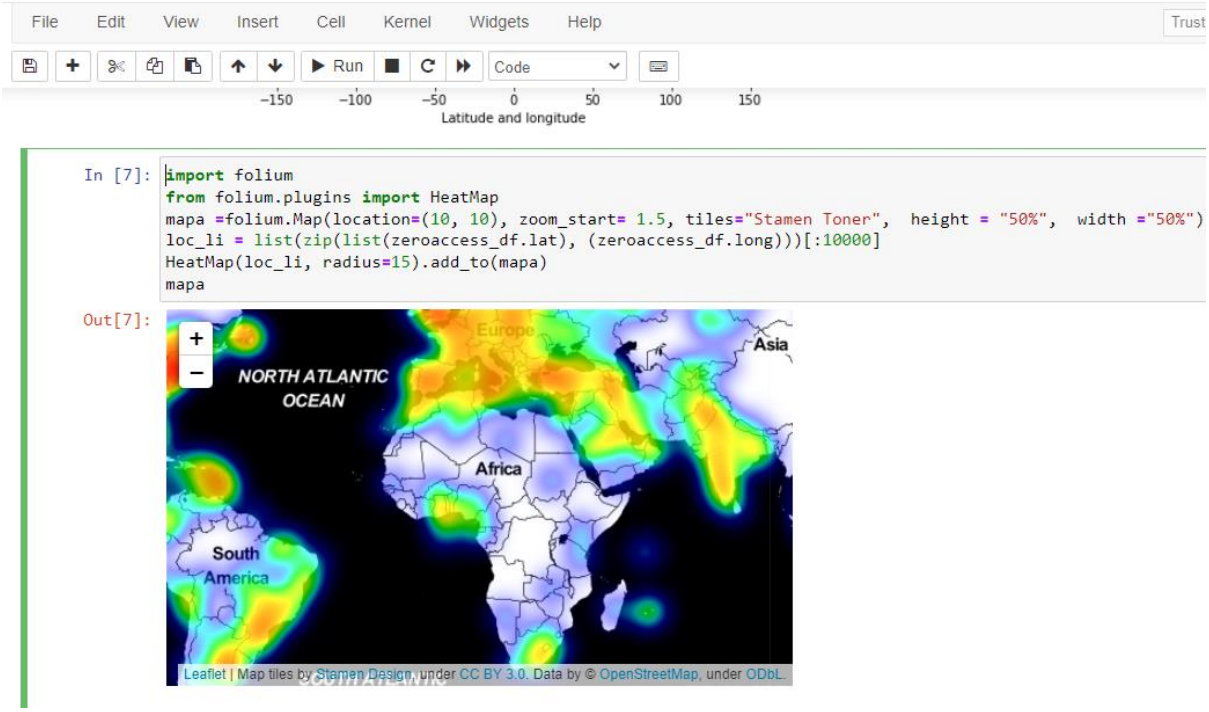
Run Code

In [6]:
```
fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111)
ax.scatter(zeroaccess_df.long, zeroaccess_df.lat, alpha = 0.1, s = 1, label = "Zero

ax.set_xlim( (-180, 180))
ax.set_ylim((-90, 90))

plt.xlabel('Latitude and longitude')
plt.ylabel('Latitude and longitude')
plt.legend(loc='upper left')
plt.show()
```

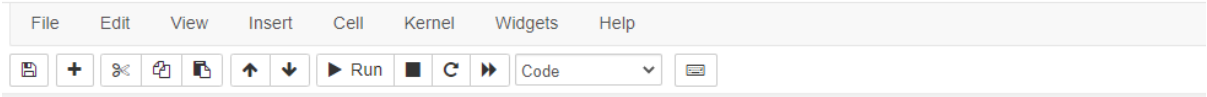
Step 4:


File Edit View Insert Cell Kernel Widgets Help
Run
Code
plt.xlabel('Latitude and longitude')
plt.ylabel('Latitude and longitude')
plt.legend(loc='upper left')
plt.show()
ZeroAccess
Latitude and longitude
Latitude and longitude

Step 5:


File Edit View Insert Cell Kernel Widgets Help
Run
Code
Latitude and longitude
In [7]:
import folium
from folium.plugins import HeatMap
mapa =folium.Map(location=(10, 10), zoom_start= 1.5, tiles="Stamen Toner", height = "50%", width ="50%")
loc_li = list(zip(list(zeroaccess_df.lat), (zeroaccess_df.long)))[:10000]
HeatMap(loc_li, radius=15).add_to(mapa)
mapa
Out[7]:
NORTH ATLANTIC OCEAN
Europe
Asia
Africa
South America
Leaflet | Map tiles by Stamen Design, under CC BY 3.0. Data by © OpenStreetMap, under ODbL.

Step 6:

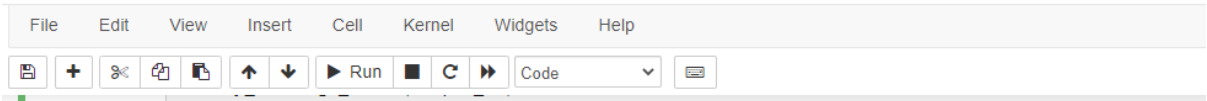


In [8]:
```
import reverse_geocoder

def rgr_batch(lat_long_df):
    cord_li = []
    for ind in lat_long_df.index:
        cord_li.append((lat_long_df['lat'][ind], lat_long_df['long'][ind]))
    result_dict_li = reverse_geocoder.search(cord_li)
    return pd.DataFrame(result_dict_li).drop(["lat", "lon"], axis = 1)

sample_df =zeroaccess_df.head(50000)
country_df = rgr_batch(sample_df)

df = sample_df.join(country_df).rename(columns={'name':'City', 'admin1':'State', 'admin2':'County',
                                                'cc': 'country_code', })
df
```

```
Loading formatted geocoded file...
```

Out[8]:

	lat	long	City	State	County	country_code
0	-10.0000	-55.0000	Alta Floresta	Mato Grosso	Alta Floresta	BR
1	38.0888	-78.5592	Charlottesville	Virginia	City of Charlottesville	US
2	38.9990	-84.6266	Florence	Kentucky	Boone County	US




```
df = sample_df.join(country_df).rename(columns={'name':'City', 'admin1':'State', 'admin2':'County',
                                                'cc': 'country_code', })
df
```

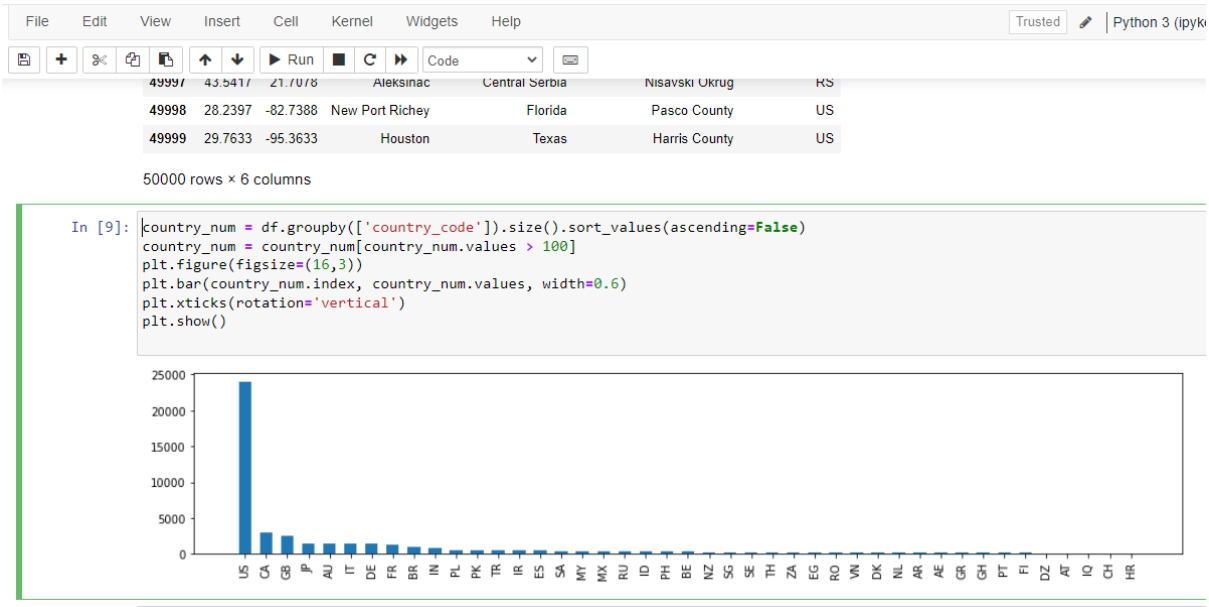
```
Loading formatted geocoded file...
```

Out[8]:

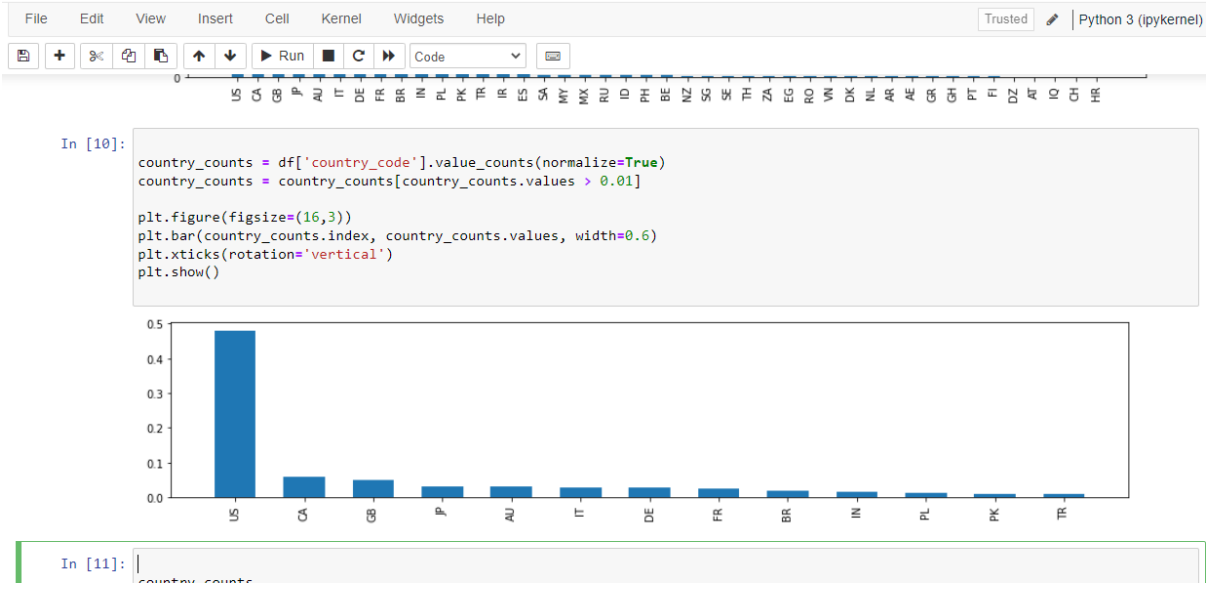
	lat	long	City	State	County	country_code
0	-10.0000	-55.0000	Alta Floresta	Mato Grosso	Alta Floresta	BR
1	38.0888	-78.5592	Charlottesville	Virginia	City of Charlottesville	US
2	38.9990	-84.6266	Florence	Kentucky	Boone County	US
3	48.6210	7.4944	Marlenheim	Alsace	Departement du Bas-Rhin	FR
4	43.2342	-86.2484	Muskegon	Michigan	Muskegon County	US
...	...	...	...	...	...	...
49995	53.1271	18.0200	Bydgoszcz	Kujawsko-Pomorskie	Bydgoszcz	PL
49996	42.6757	-82.7773	New Baltimore	Michigan	Macomb County	US
49997	43.5417	21.7078	Aleksinac	Central Serbia	Nisavski Okrug	RS
49998	28.2397	-82.7388	New Port Richey	Florida	Pasco County	US
49999	29.7633	-95.3633	Houston	Texas	Harris County	US

50000 rows × 6 columns

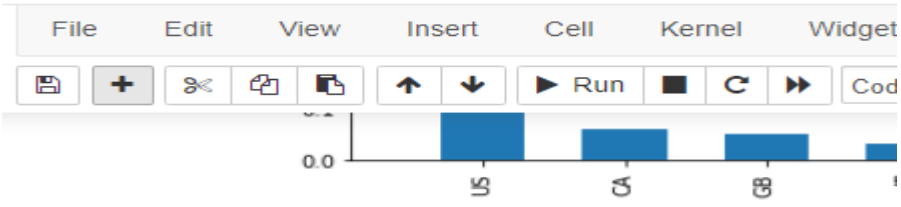
Step 7:


File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipyk
Run
Code
49997 43.5417 21.7078 Aleksinac Central Serbia Nisavski Okrug RS
49998 28.2397 -82.7388 New Port Richey Florida Pasco County US
49999 29.7633 -95.3633 Houston Texas Harris County US
50000 rows × 6 columns
In [9]:
country_num = df.groupby(['country_code']).size().sort_values(ascending=False)
country_num = country_num[country_num.values > 100]
plt.figure(figsize=(16,3))
plt.bar(country_num.index, country_num.values, width=0.6)
plt.xticks(rotation='vertical')
plt.show()
25000
20000
15000
10000
5000
0
US CA GB JP AU IT DE FR BR IN PL PK TR IR ES SA MY MX RU ID PH BE NZ SG SE TH ZA EG RO VN DK NL AR AE GR GH PT FI DZ AT IQ CH HR

Step 8:

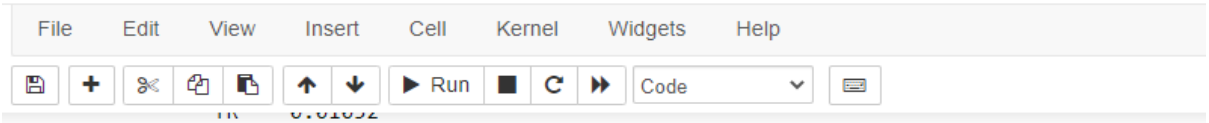

File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
0
US CA GB JP AU IT DE FR BR IN PL PK TR IR ES SA MY MX RU ID PH BE NZ SG SE TH ZA EG RO VN DK NL AR AE GR GH PT FI DZ AT IQ CH HR
In [10]:
country_counts = df['country_code'].value_counts(normalize=True)
country_counts = country_counts[country_counts.values > 0.01]
plt.figure(figsize=(16,3))
plt.bar(country_counts.index, country_counts.values, width=0.6)
plt.xticks(rotation='vertical')
plt.show()
0.5
0.4
0.3
0.2
0.1
0.0
US CA GB JP AU IT DE FR BR IN PL PK TR
In [11]:

Step 9:




```
In [11]: country_counts

Out[11]: US    0.47974
         CA    0.05932
         GB    0.05128
         JP    0.03008
         AU    0.03002
         IT    0.02964
         DE    0.02824
         FR    0.02622
         BR    0.01928
         IN    0.01496
         PL    0.01154
         PK    0.01096
         TR    0.01052
         Name: country_code, dtype: float64
```

Step 10:




```
Name: country_code, dtype: float64
```

```
In [12]: us_df = df[df['country_code']=='US']
         us_df
```

Out[12]:

	lat	long	City	State	County	country_code
1	38.0888	-78.5592	Charlottesville	Virginia	City of Charlottesville	US
2	38.9990	-84.6266	Florence	Kentucky	Boone County	US
4	43.2342	-86.2484	Muskegon	Michigan	Muskegon County	US
6	44.8012	-68.7778	Bangor	Maine	Penobscot County	US
8	36.0580	-88.8253	Bradford	Tennessee	Gibson County	US
...	...	...	...	...	...	...
49992	40.6374	-79.7264	Brackenridge	Pennsylvania	Allegheny County	US
49994	38.4380	-122.6753	Santa Rosa	California	Sonoma County	US
49996	42.6757	-82.7773	New Baltimore	Michigan	Macomb County	US
49998	28.2397	-82.7388	New Port Richey	Florida	Pasco County	US
49999	29.7633	-95.3633	Houston	Texas	Harris County	US

23987 rows × 6 columns

Step 11:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [13]:
```
fig = plt.figure(figsize=(8,6))
ax = fig.add_subplot(111)

ax.scatter(us_df.long, us_df.lat, alpha = 0.1, s = 1)
plt.show()
```

Step 12:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [14]:
```
df.State.value_counts()
[df.State.value_counts()>30]
```

Out[14]:
```
[England              True
 California           True
 Texas                True
 Florida              True
 Kansas               True
                      ...
 Orange Free State    False
 Bihar                False
 South East           False
 Tombouctou           False
 Smolensk             False
 Name: State, Length: 1178, dtype: bool]
```

In [15]:
```
df.groupby(['State','City']).size().sort_values(ascending=False)
```

Out[15]:
```
State              City
Kansas             Peabody                1152
England            London                  453
Tokyo              Tokyo                   427
Islamabad          Islamabad               319
England            City of London          300
```

File Edit View Insert Cell Kernel Widgets Help

Run Code

South East   False
Tombouctou   False
Smolensk   False
Name: State, Length: 1178, dtype: bool]

In [15]:
df.groupby(['State','City']).size().sort_values(ascending=False)

Out[15]:
```
State              City
Kansas             Peabody                1152
England            London                  453
Tokyo              Tokyo                   427
Islamabad          Islamabad               319
England            City of London          300
                                           ...
Nord-Pas-de-Calais Lens                      1
                   Lambersart                1
                   Lallaing                  1
                   La Bassee                 1
Midi-Pyrenees      Montesquieu-Volvestre     1
Length: 13138, dtype: int64
```

Step 13:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [25]:

```
county_df = pd.read_csv("county-data.csv")
county = county_df.groupby('subregion')['pop','income','ipaddr','ufo2010'].apply(sum)


us_count = us_df.groupby('State').size().sort_values(ascending=False).reset_index()
us_count.columns = ['state','attack']
us_count['att_pc'] = us_count['attack']/us_count.attack.sum()
us_count = us_count.set_index(us_count['state'])
us_count.drop(['state'],axis=1,inplace=True)

us_pop = state_df[['state','population']]
us_pop['pop_pc'] = state_df.population/state_df.population.sum()
us_pop = us_pop.set_index(us_pop['state'])
us_pop.drop(['state'],axis=1,inplace=True)


att_pop = pd.concat([us_count['attack'], us_count['att_pc'], us_pop['pop_pc']], axis=1, sort=False)

att_pop.sort_index()
lower_index =[each.lower() for each in att_pop.index]
att_pop = att_pop.set_index(pd.Index(lower_index))

county_att_pop = county.merge(att_pop, left_index=True, right_index=True)
county_att_pop
```

File Edit View Insert Cell Kernel Widgets Help

Run Code

Out[25]:

	pop	income	ipaddr	ufo2010	attack	att_pc	pop_pc
arkansas	18892	38986	9733	0	262.0	0.010923	0.009473
colorado	20696	43252	8890	3	394.0	0.016426	0.016474
delaware	965814	320910	1274228	60	68.0	0.002835	0.002902
idaho	16308	36706	13280	6	99.0	0.004127	0.005068
indiana	88218	41424	101611	10	458.0	0.019094	0.021058
iowa	39996	113101	30522	1	228.0	0.009505	0.009861
mississippi	59884	62741	13601	3	241.0	0.010047	0.009678
nevada	107217	96083	75150	11	187.0	0.007796	0.008665
new york	1619090	67204	12977872	6	1315.0	0.054821	0.064068
ohio	74229	130235	115211	3	914.0	0.038104	0.037843
oklahoma	741781	44413	1156716	72	293.0	0.012215	0.012088
oregon	10997	27885	1321	2	235.0	0.009797	0.012543
texas	47308	79759	25136	9	2003.0	0.083504	0.081250
utah	540504	59338	2074111	28	164.0	0.006837	0.009129
washington	3193424	1514569	15239854	241	502.0	0.020928	0.021849
wyoming	93290	136016	54251	3	38.0	0.001584	0.001784

Step 14:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [26]:
```
import numpy as np
corr_matrix = county_att_pop.corr()
corr_matrix.style.background_gradient()
```

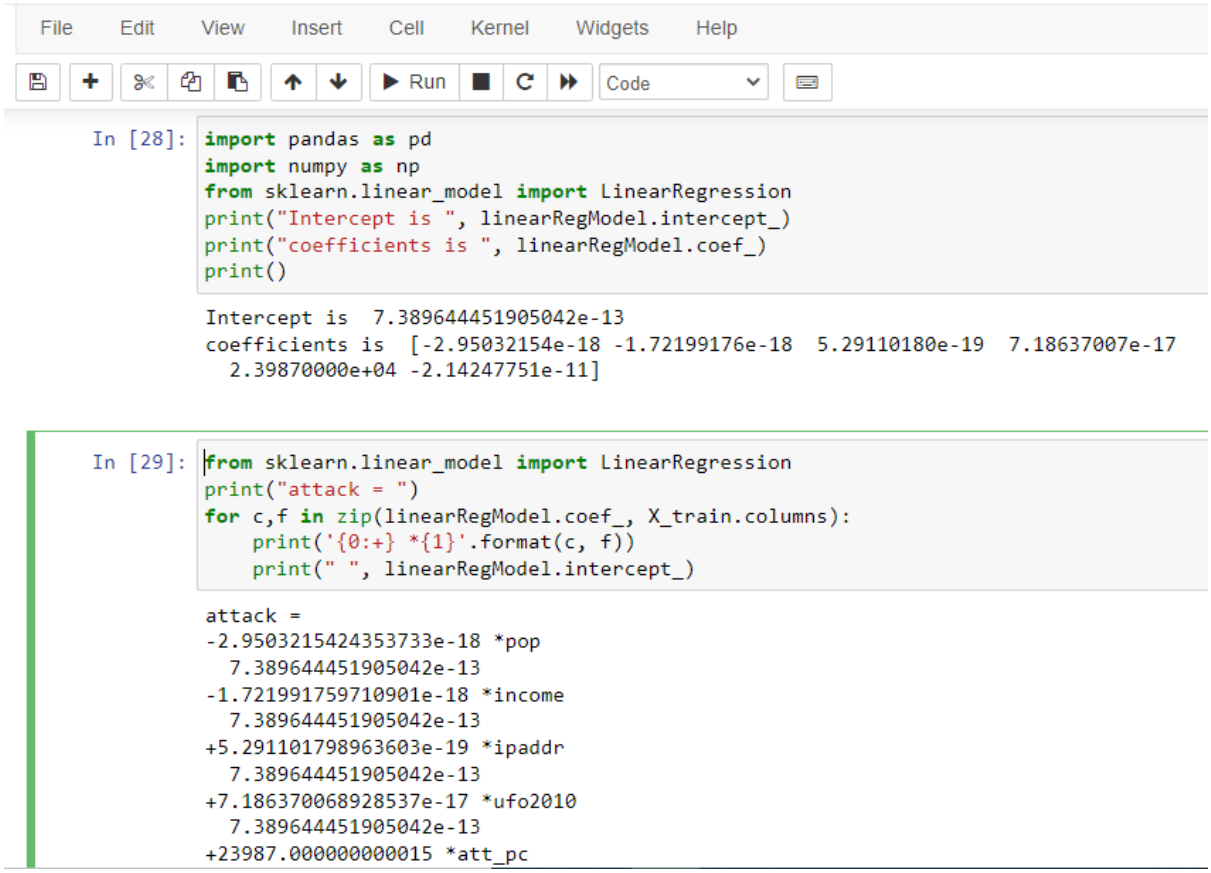
Out[26]:

	pop	income	ipaddr	ufo2010	attack	att_pc	pop_pc
pop	1.000000	0.861982	0.934995	0.888594	0.123880	0.123880	0.163024
income	0.861982	1.000000	0.722039	0.946220	-0.002957	-0.002957	-0.006568
ipaddr	0.934995	0.722039	1.000000	0.699879	0.275919	0.275919	0.337161
ufo2010	0.888594	0.946220	0.699879	1.000000	-0.047008	-0.047008	-0.049894
attack	0.123880	-0.002957	0.275919	-0.047008	1.000000	1.000000	0.993622
att_pc	0.123880	-0.002957	0.275919	-0.047008	1.000000	1.000000	0.993622
pop_pc	0.163024	-0.006568	0.337161	-0.049894	0.993622	0.993622	1.000000

In [27]:
```
import pandas as pd
import numpy as np
X_train = county_att_pop.drop(['attack'], axis=1)
y_train = county_att_pop['attack']
from sklearn.linear_model import LinearRegression
linearRegModel = LinearRegression()
linearRegModel.fit(X_train, y_train)
```

Out[27]:
```
LinearRegression()
```

Step 15:


File Edit View Insert Cell Kernel Widgets Help
Run
Code
In [28]:
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
print("Intercept is ", linearRegModel.intercept_)
print("coefficients is ", linearRegModel.coef_)
print()
Intercept is 7.389644451905042e-13
coefficients is [-2.95032154e-18 -1.72199176e-18 5.29110180e-19 7.18637007e-17
2.39870000e+04 -2.14247751e-11]
In [29]:
from sklearn.linear_model import LinearRegression
print("attack = ")
for c,f in zip(linearRegModel.coef_, X_train.columns):
print('{0:+} *{1}'.format(c, f))
print(" ", linearRegModel.intercept_)
attack =
-2.9503215424353733e-18 *pop
7.389644451905042e-13
-1.721991759710901e-18 *income
7.389644451905042e-13
+5.291101798963603e-19 *ipaddr
7.389644451905042e-13
+7.186370068928537e-17 *ufo2010
7.389644451905042e-13
+23987.000000000015 *att_pc

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
  7.389644451905042e-13
-2.1424775099086808e-11 *pop_pc
  7.389644451905042e-13
```

In [30]:
```
import statsmodels.api as sm
import warnings
warnings.filterwarnings('ignore')
results = sm.OLS(y_train, X_train).fit()
results.summary()
```

Out[30]:

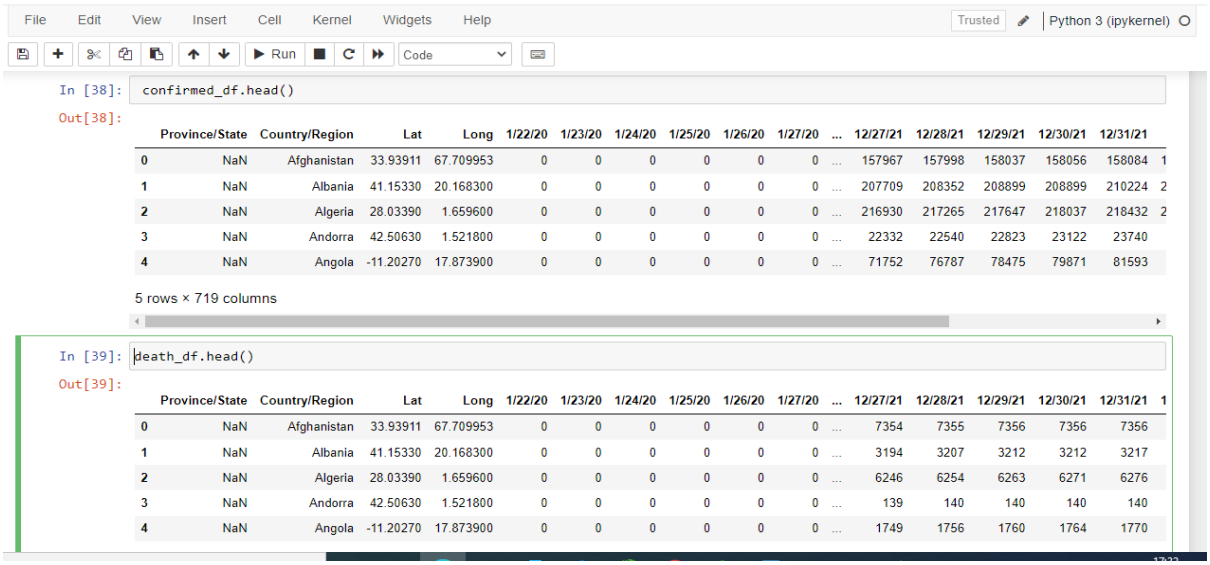
OLS Regression Results

Dep. Variable:	attack	**R-squared (uncentered):**	1.000
Model:	OLS	**Adj. R-squared (uncentered):**	1.000
Method:	Least Squares	**F-statistic:**	1.442e+30
Date:	Sun, 09 Jan 2022	**Prob (F-statistic):**	4.33e-149
Time:	23:00:54	**Log-Likelihood:**	424.20
No. Observations:	16	**AIC:**	-836.4
Df Residuals:	10	**BIC:**	-831.8
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Covariance Type:	nonrobust					
pop	2.06e-18	2.16e-18	0.953	0.363	-2.76e-18	6.88e-18
income	2.548e-18	2.33e-18	1.095	0.299	-2.64e-18	7.73e-18
ipaddr	-2.507e-19	2.44e-19	-1.027	0.328	-7.94e-19	2.93e-19
ufo2010	-2.132e-14	2.32e-14	-0.919	0.380	-7.3e-14	3.04e-14
att_pc	2.399e+04	1.96e-10	1.23e+14	0.000	2.4e+04	2.4e+04
pop_pc	-7.276e-12	1.97e-10	-0.037	0.971	-4.46e-10	4.32e-10

Omnibus:	4.572	**Durbin-Watson:**	1.091
Prob(Omnibus):	0.102	**Jarque-Bera (JB):**	2.746
Skew:	-1.011	**Prob(JB):**	0.253
Kurtosis:	3.178	**Cond. No.**	6.09e+09

Step 16:



File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [38]:
```
confirmed_df.head()
```

Out[38]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	12/27/21	12/28/21	12/29/21	12/30/21	12/31/21	
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	157967	157998	158037	158056	158084	1
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	207709	208352	208899	208899	210224	2
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...	216930	217265	217647	218037	218432	2
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	...	22332	22540	22823	23122	23740	
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	...	71752	76787	78475	79871	81593	

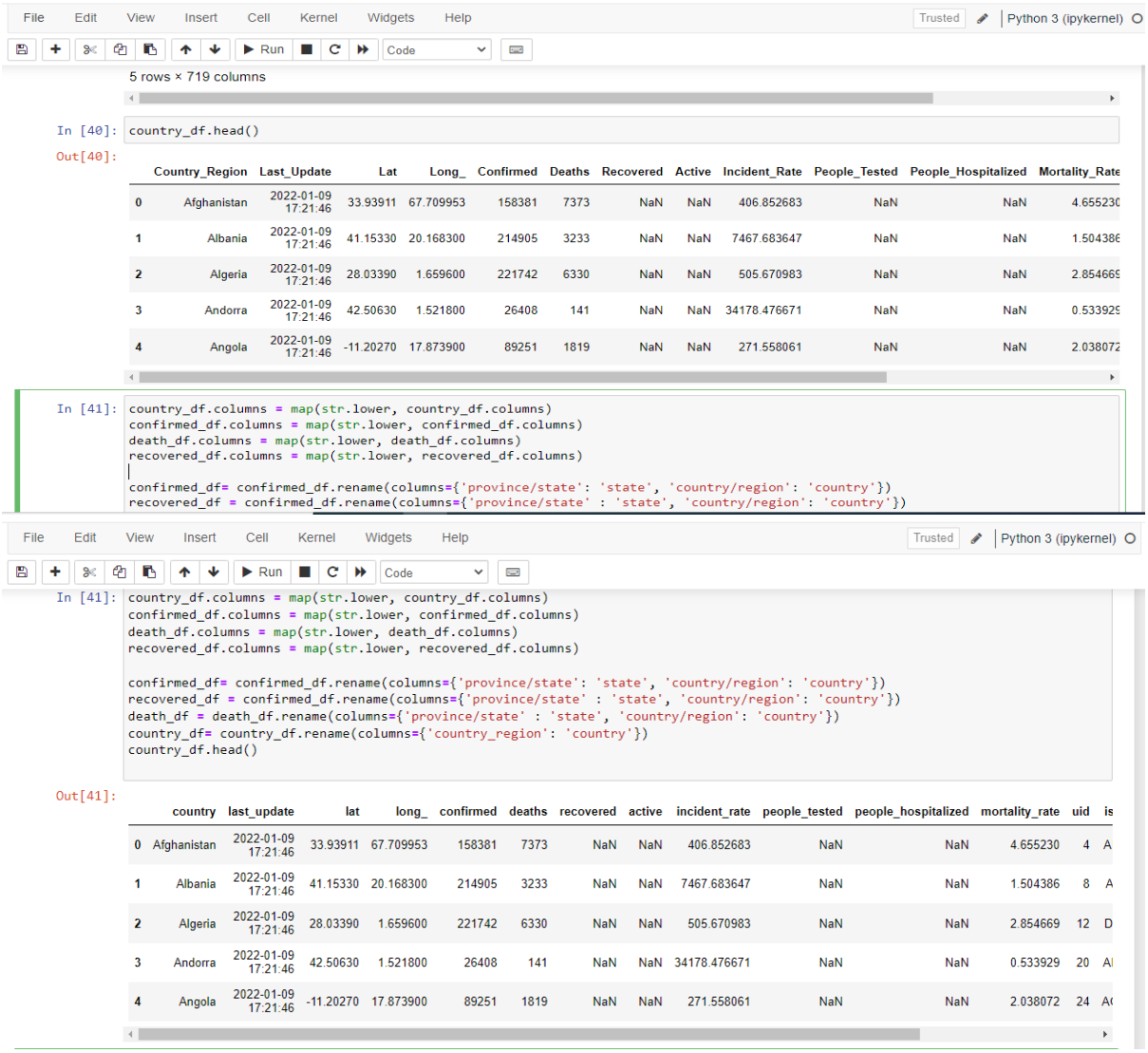
5 rows × 719 columns

In [39]:
```
death_df.head()
```

Out[39]:

	Province/State	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	...	12/27/21	12/28/21	12/29/21	12/30/21	12/31/21	1
0	NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	...	7354	7355	7356	7356	7356	
1	NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	...	3194	3207	3212	3212	3217	
2	NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	...	6246	6254	6263	6271	6276	
3	NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	...	139	140	140	140	140	
4	NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	...	1749	1756	1760	1764	1770	

Step 17:



File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

5 rows × 719 columns

In [40]:
```
country_df.head()
```

Out[40]:

	Country_Region	Last_Update	Lat	Long_	Confirmed	Deaths	Recovered	Active	Incident_Rate	People_Tested	People_Hospitalized	Mortality_Rate
0	Afghanistan	2022-01-09 17:21:46	33.93911	67.709953	158381	7373	NaN	NaN	406.852683	NaN	NaN	4.655230
1	Albania	2022-01-09 17:21:46	41.15330	20.168300	214905	3233	NaN	NaN	7467.683647	NaN	NaN	1.504386
2	Algeria	2022-01-09 17:21:46	28.03390	1.659600	221742	6330	NaN	NaN	505.670983	NaN	NaN	2.854669
3	Andorra	2022-01-09 17:21:46	42.50630	1.521800	26408	141	NaN	NaN	34178.476671	NaN	NaN	0.533929
4	Angola	2022-01-09 17:21:46	-11.20270	17.873900	89251	1819	NaN	NaN	271.558061	NaN	NaN	2.038072

In [41]:
```
country_df.columns = map(str.lower, country_df.columns)
confirmed_df.columns = map(str.lower, confirmed_df.columns)
death_df.columns = map(str.lower, death_df.columns)
recovered_df.columns = map(str.lower, recovered_df.columns)

confirmed_df= confirmed_df.rename(columns={'province/state': 'state', 'country/region': 'country'})
recovered_df = confirmed_df.rename(columns={'province/state' : 'state', 'country/region': 'country'})
```


File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

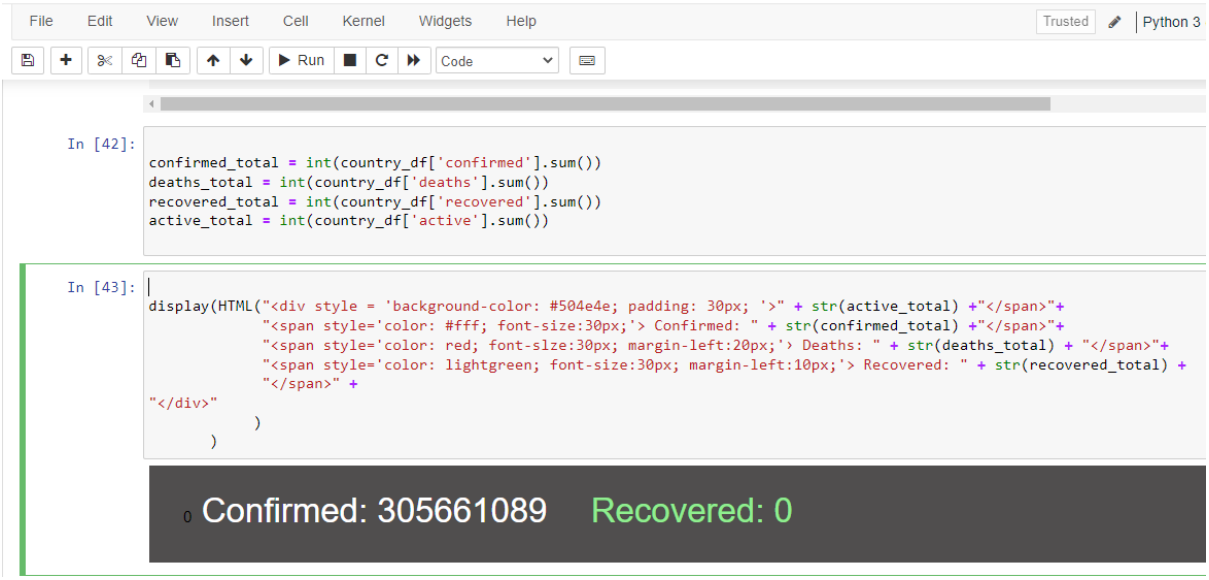
Run Code

In [41]:
```
country_df.columns = map(str.lower, country_df.columns)
confirmed_df.columns = map(str.lower, confirmed_df.columns)
death_df.columns = map(str.lower, death_df.columns)
recovered_df.columns = map(str.lower, recovered_df.columns)

confirmed_df= confirmed_df.rename(columns={'province/state': 'state', 'country/region': 'country'})
recovered_df = confirmed_df.rename(columns={'province/state' : 'state', 'country/region': 'country'})
death_df = death_df.rename(columns={'province/state' : 'state', 'country/region': 'country'})
country_df= country_df.rename(columns={'country_region': 'country'})
country_df.head()
```

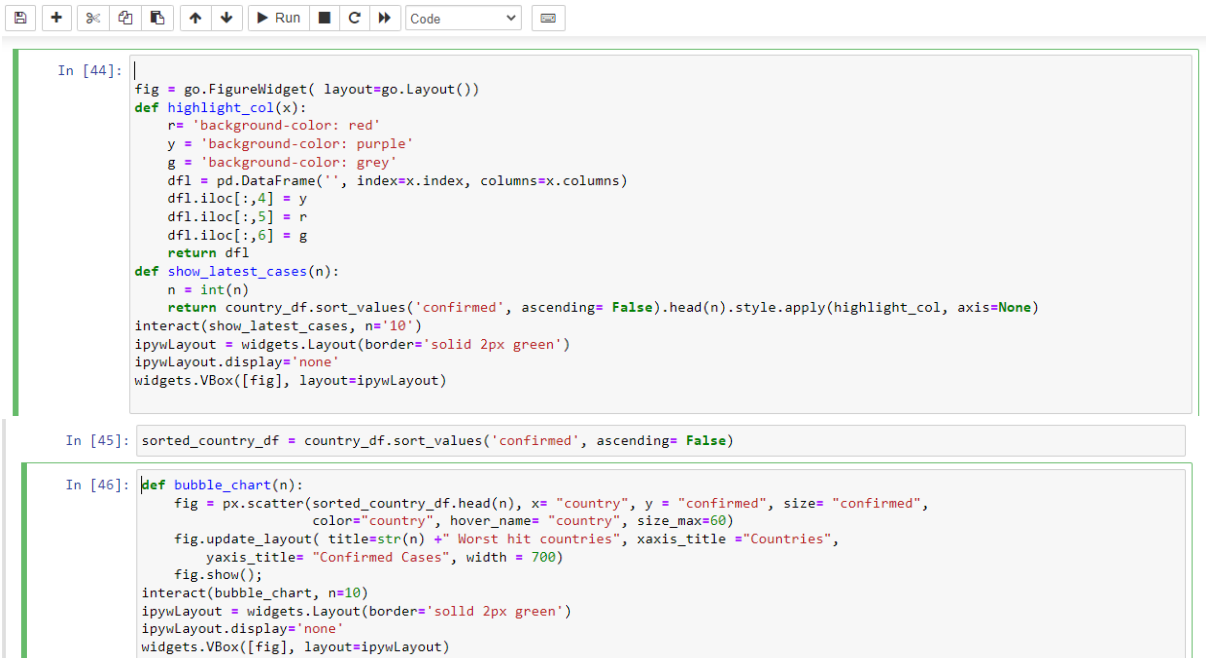
Out[41]:

	country	last_update	lat	long_	confirmed	deaths	recovered	active	incident_rate	people_tested	people_hospitalized	mortality_rate	uid	is
0	Afghanistan	2022-01-09 17:21:46	33.93911	67.709953	158381	7373	NaN	NaN	406.852683	NaN	NaN	4.655230	4	A
1	Albania	2022-01-09 17:21:46	41.15330	20.168300	214905	3233	NaN	NaN	7467.683647	NaN	NaN	1.504386	8	A
2	Algeria	2022-01-09 17:21:46	28.03390	1.659600	221742	6330	NaN	NaN	505.670983	NaN	NaN	2.854669	12	D
3	Andorra	2022-01-09 17:21:46	42.50630	1.521800	26408	141	NaN	NaN	34178.476671	NaN	NaN	0.533929	20	Al
4	Angola	2022-01-09 17:21:46	-11.20270	17.873900	89251	1819	NaN	NaN	271.558061	NaN	NaN	2.038072	24	Ad

Step 18:



Step 19:



File Edit View Insert Cell Kernel Widgets Help Trusted

Run Code

In [48]:
```
def plot_cases_of_a_country(country):
    labels = ['confirmed', 'deaths']
    colors = ['blue', 'red']
    mode_size = [6, 8]
    line_size = [4, 5]

    df_list = [confirmed_df, death_df]

    fig = go.Figure();

    for i, df in enumerate(df_list):
        if country == 'World' or country == 'world':
            x_data = np.array(list(df.iloc[:, 20:].columns))
            y_data = np.sum(np.asarray(df.iloc[:,4:]),axis = 0)

        else:
            x_data = np.array(list(df.iloc[:, 20:].columns))
            y_data = np.sum(np.asarray(df[df['country'] == country].iloc[:,20:]),axis = 0)
        print(i)
        fig.add_trace(go.Scatter(x=x_data, y=y_data, mode='lines+markers',
        name=labels[i],
        line=dict(color=colors[i], width=line_size[i]),
        connectgaps=True,
        text = "Total " + str(labels[i]) +": "+ str(y_data[-1])
        ));

    fig.update_layout(
```

File Edit View Insert Cell Kernel Widgets Help

Run Code

```
        ));

    fig.update_layout(
        title="COVID 19 cases of " + country,
        xaxis_title='Date',
        yaxis_title='No. of Confirmed Cases',
        margin=dict(l=20, r=20, t=40, b=20),
        paper_bgcolor="lightgrey",
        width = 800,

    );

    fig.update_yaxes(type="linear")
    fig.show();
```

In [49]:
```
interact(plot_cases_of_a_country, country='World')

ipywLayout = widgets.Layout(border='solid 2px green')
ipywLayout.display='none'
widgets.VBox([fig], layout=ipywLayout)
```

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [51]:
```
px.bar(
    sorted_country_df.head(10),
    x = "country",
    y = "confirmed",
    title= "Top 10 worst affected countries", # the axis names
    color_discrete_sequence=["pink"],
    height=500,
    width=800
)
```

Step 20:

File Edit View Insert Cell Kernel Widgets Help Trusted

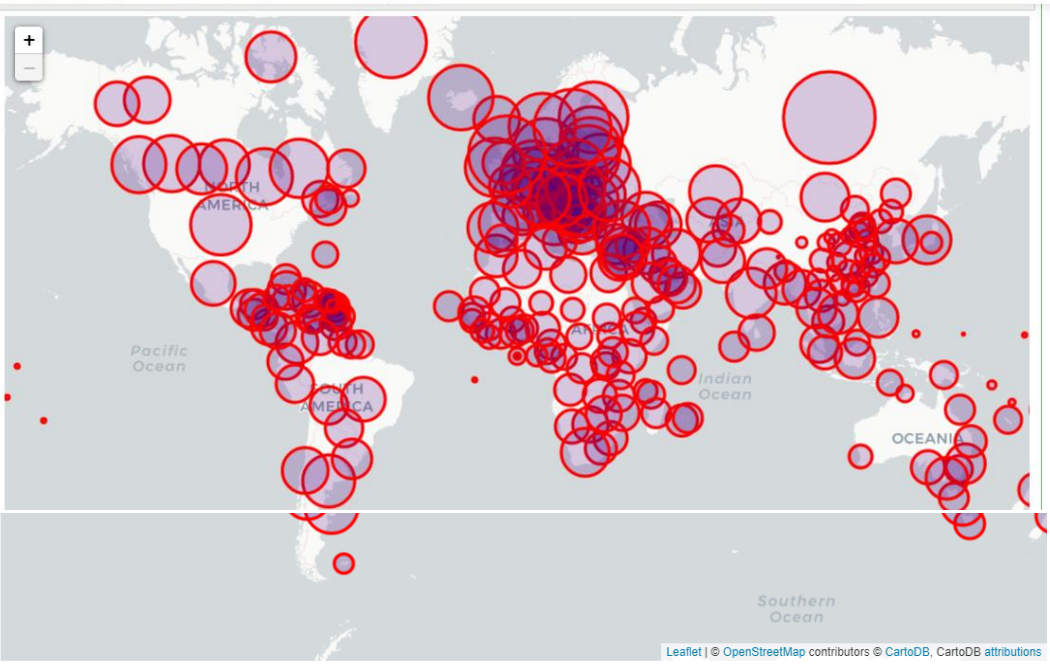
Run Code

In [53]:
```
confirmed_df = confirmed_df.dropna(subset=['long'])
confirmed_df = confirmed_df.dropna(subset=['lat'])

world_map = folium.Map(location=[11,0], tiles="cartodbpositron",zoom_start=2 , max_zoom = 6, min_zoom = 2)

for i in range(0, len(confirmed_df)):
    folium.Circle(
    location=[confirmed_df.iloc[i]['lat'], confirmed_df.iloc[i]['long']],
        fill= True,
        radius=(int((np.log(confirmed_df.iloc[i,-1]+1.00001)))+0.2)*50000,
        color='red',
        fill_color='indigo',
        tooltip = "<div style='margin: 0; background-color: black; color: white'>"+
        "<h4 style=stext-align:center;font-weight: bold'>"+confirmed_df.iloc[i]['country'] + "</h4>"
        "<hr style='margin:10px;color: white'>"+
        "<ul style='color: white;;list-style-type:circle;align-item:left;padding-left:20px;padding-right:20px'>"+
        "<li>Confirmed: "+str(confirmed_df.iloc[i,-1])+"</li>"+
        "<li>Deaths: "+str(death_df.iloc[i,-1])+"</li>"+
        "<li>Death Rate: "+ str(np.round(death_df.iloc[i,-1]/(confirmed_df.iloc[i,-1]+1.00001)*100,2))+
"</li> " +
"</u1></div>", ).add_to(world_map)
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

Out[53]:



Part 3: Challenging Questions

Question-1:

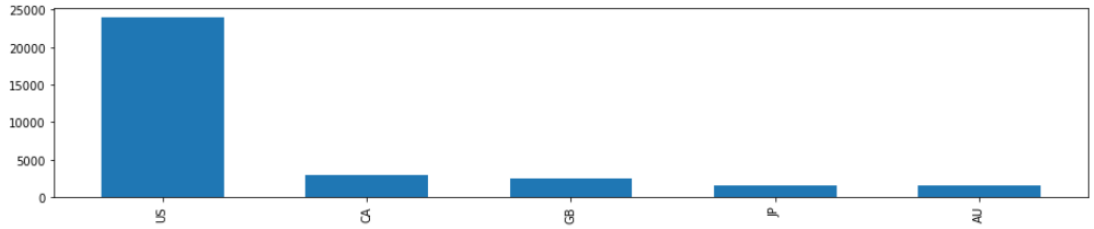
Answer: The zero access takes place in the United States as we can see from the figures that more attacks are reported in the United States. The attack is mostly targeted towards larger population countries like the US and Canada. Because they have high income rates as this attack demands money.

Question-2:

Answer: Yes, the users who do not know about this type of threat can be easily affected. As we can see, education and income affect the number of infections. The countries which have high income and education are more prone to the attack. From the heat map graph and from the regression model we have seen that

Question-3:

```
In [54]: import plotly.express as px
country_num= df.groupby(['country_code']).size().sort_values(ascending=False)
country_num= country_num[country_num.values > 100]
plt.figure(figsize=(16,3))
plt.bar(country_num.index[0:5], country_num.values[0:5], width=0.6)
plt.xticks(rotation='vertical')
plt.show()
```



Question-4:

Answer:
In Kansas state there is a city named Peabody which has the highest rate of infection attack. The geographic center is outside the Potwin, a place near to Kansas and it is referred to as Unknown locations in the US.

Question 5:

File Edit View Insert Cell Kernel Widgets Help

Run Code

In [56]:
```
us_count = us_df.groupby('State').size().sort_values(ascending=False).reset_index()
us_count.columns = ['state','attack']
us_count['att_pc'] = us_count['attack'] / us_count.attack.sum()
us_count = us_count.set_index(us_count['state'])
us_count.drop(['state'],axis=1,inplace=True)
us_count
```

Out[56]:

state	attack	att_pc
California	2293	0.095593
Texas	2003	0.083504
Florida	1425	0.059407
Kansas	1371	0.057156
New York	1315	0.054821
Pennsylvania	970	0.040439
Illinois	948	0.039521
Ohio	914	0.038104
Georgia	841	0.035061
North Carolina	759	0.031642
Michigan	715	0.029808
New Jersey	669	0.027890
Virginia	526	0.021929
Arizona	503	0.020970
Washington	502	0.020928
Missouri	480	0.020011
Wisconsin	472	0.019677
Indiana	458	0.019094
Massachusetts	453	0.018885
Maryland	430	0.017926
Tennessee	425	0.017718
Alabama	412	0.017176
Minnesota	405	0.016884
Colorado	394	0.016426
South Carolina	386	0.016092
Kentucky	373	0.015550
Louisiana	366	0.015258
Oklahoma	293	0.012215

Arkansas	262	0.010923
Mississippi	241	0.010047
Oregon	235	0.009797
Iowa	228	0.009505
Connecticut	226	0.009422
Nevada	187	0.007796
Utah	164	0.006837
West Virginia	161	0.006712
Nebraska	144	0.006003
New Mexico	133	0.005545
Maine	113	0.004711
Idaho	99	0.004127
New Hampshire	98	0.004086
Montana	83	0.003460
Rhode Island	77	0.003210
Hawaii	73	0.003043
South Dakota	69	0.002877

Answer: The output from the above code shows the attack percentage for each state in the US

File Edit View Insert Cell Kernel Widgets Help

Run Markdown

In [59]:
```
us_pop = state_df[['state','population']]
us_pop['pop_pc'] = state_df.population/state_df.population.sum()
us_pop = us_pop.set_index(us_pop['state'])
us_pop.drop(['state'],axis=1,inplace=True)
us_pop
```

Out[59]:

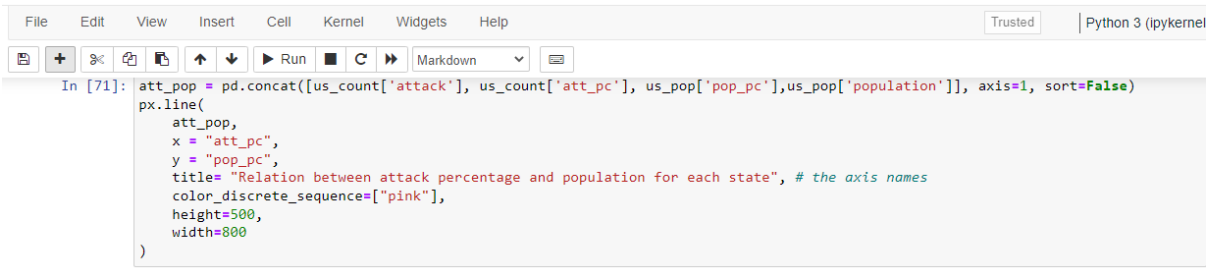
state	population	pop_pc
Alabama	4758191	0.015438
Arizona	6665093	0.021625
Arkansas	2919815	0.009473
California	37350092	0.121181
Colorado	5077553	0.016474
Connecticut	3555261	0.011535
Delaware	894424	0.002902
District of Columbia	605959	0.001966
Florida	18732783	0.060778
Georgia	9932505	0.032226
Idaho	1562046	0.005068
Indiana	6490613	0.021058
Iowa	3039465	0.009861
Kansas	2848369	0.009241
Kentucky	4359450	0.014144
Louisiana	4539283	0.014727
Maine	1332155	0.004322
Maryland	5759373	0.018686
Massachusetts	6662878	0.021617
Michigan	10074498	0.032686
Minnesota	5321556	0.017266
Mississippi	2983018	0.009678
Missouri	6050503	0.019631
Montana	985235	0.003197
Nebraska	1815500	0.005890
Nevada	2670861	0.008665
New Hampshire	1338495	0.004343
New Jersey	8799248	0.028549
New Mexico	2030790	0.006589

New York	19746813	0.064068
North Carolina	9479467	0.030756
North Dakota	653642	0.002121
Ohio	11663946	0.037843
Oklahoma	3725797	0.012088
Oregon	3865861	0.012543
Pennsylvania	12737230	0.041325
Rhode Island	1064277	0.003453
South Carolina	4609176	0.014954
South Dakota	820920	0.002663
Tennessee	6362421	0.020643
Texas	25042738	0.081250
Utah	2813835	0.009129
Vermont	628294	0.002038
Virginia	7965428	0.025843
Washington	6734229	0.021849
West Virginia	1838901	0.005966

Washington	6734229	0.021849
West Virginia	1838901	0.005966
Wisconsin	5714200	0.018539
Wyoming	549990	0.001784

Answer: The above output from the code shows percentage of population which is attacked in each state of the US.

Question-6:



Question-7:

In this question as we have seen there is no value for recovered attribute in the dataset. So, I have plot only deaths for worst 20 countries hit by COVID-19