In [1]:
```
import os
from google.cloud import bigquery
import pandas as pd


# ***I commented the following imports out but you will likely need at least so
#import matplotlib.pyplot as plt
#import numpy as np
#import scipy
#from scipy.stats import norm


# dont forget to replace <andreasfreund> with your local username
#os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = "/Users/Rebecca/.config/gcloud/
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = "/Users/Rebecca/.config/gcloud/a
%load_ext google.cloud.bigquery
client = bigquery.Client()

print('Done!')
```

```
Done!
```

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
import scipy
from scipy.stats import norm
```

National deployment data: Num of blocks that have deployment

In [3]:
```
%%bigquery num_2015_nation_table
SELECT count(distinct BlockCode) as num_2015_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2015_v5`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.04s: 100%|██████████| 1/1 [00:00<00:00, 741.83query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.79s/row
s]
```

In [4]:
```
num_2015_nation_table
```

Out[4]:

	num_2015_nation
0	5803476

In [5]:
```
%%bigquery num_2016_nation_table
SELECT count(distinct BlockCode) as num_2016_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2016_v4`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.00s: 100%|██████████| 1/1 [00:00<00:00, 172.70query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.92s/row
s]
```

In [6]:
```
num_2016_nation_table
```

Out[6]:

	num_2016_nation
0	6237057

In [7]:
```
%%bigquery num_2017_nation_table
SELECT count(distinct BlockCode) as num_2017_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2017_v3`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.01s: 100%|██████████| 1/1 [00:00<00:00, 370.78query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.65s/row
s]
```

In [8]:
```
num_2017_nation_table
```

Out[8]:

	num_2017_nation
0	6660284

In [9]:
```
type(num_2017_nation_table)
```

Out[9]:
```
pandas.core.frame.DataFrame
```

In [10]:
```
%%bigquery num_2018_nation_table
SELECT count(distinct BlockCode) as num_2018_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2018_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.01s: 100%|██████████| 1/1 [00:00<00:00, 289.76query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.44s/row
s]
```

In [11]:
```
num_2018_nation_table
```

Out[11]:

	num_2018_nation
0	7000076

In [12]:
```
%%bigquery num_2019_nation_table
SELECT count(distinct BlockCode) as num_2019_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2019_v2`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.00s: 100%|██████████| 1/1 [00:00<00:00, 364.98query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.93s/row
s]
```

In [13]:
```
num_2019_nation_table
```

Out[13]:

	num_2019_nation
0	7459404

In [14]:
```
%%bigquery num_2020_nation_table
SELECT count(distinct BlockCode) as num_2020_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2020_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.00s: 100%|██████████| 1/1 [00:00<00:00, 423.54query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.75s/row
s]
```

In [15]:
```
num_2020_nation_table
```

Out[15]:

	num_2020_nation
0	7942631

In [16]:
```
%%bigquery num_2021_nation_table
SELECT count(distinct BlockCode) as num_2021_nation
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2021_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
```

```
Query complete after 0.00s: 100%|██████████| 1/1 [00:00<00:00, 371.08query/
s]
Downloading: 100%|██████████| 1/1 [00:01<00:00,  1.70s/row
s]
```

In [17]:
```
num_2021_nation_table
```

Out[17]:

	num_2021_nation
0	8530032

In [18]:
```
# Todo: Combine the above 7 years data and make a nation level plot of num of b
```

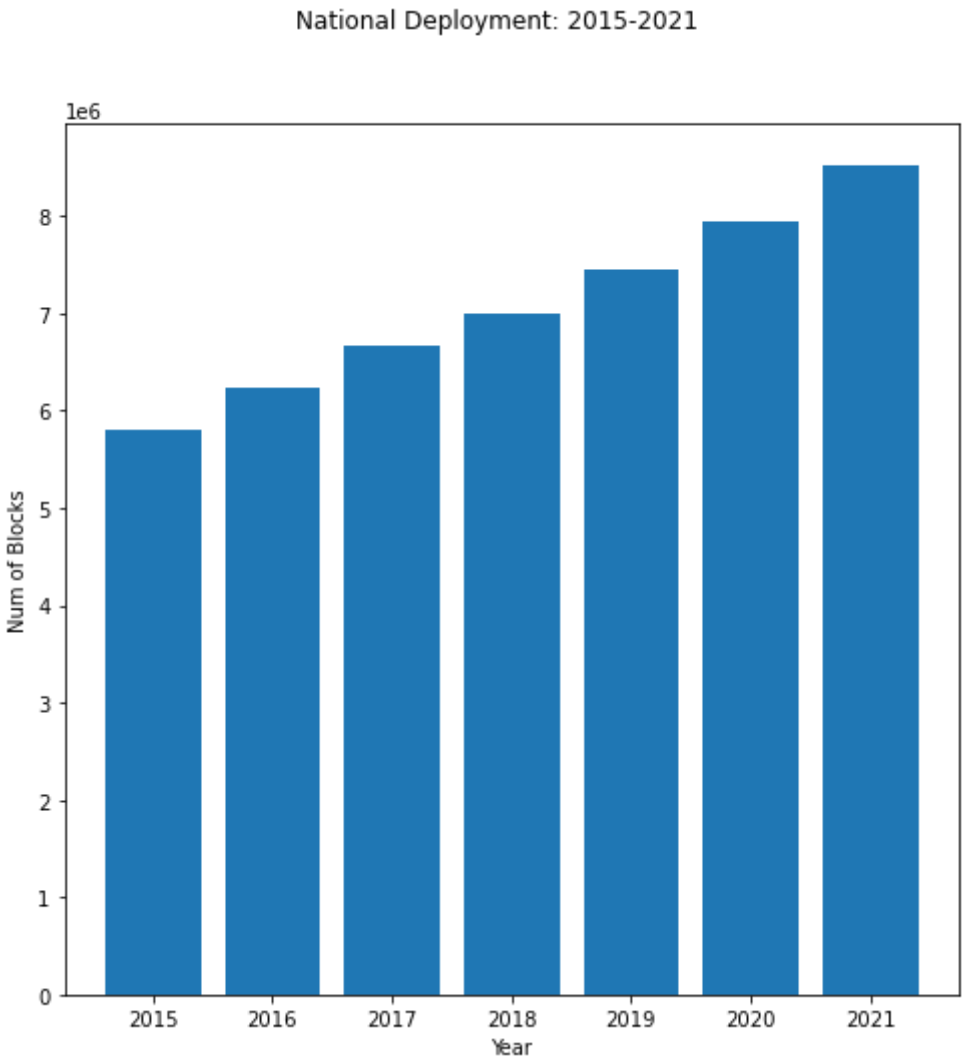
In [19]:
```
num_2015_nation_table['num']=num_2015_nation_table['num_2015_nation']
num_2016_nation_table['num']=num_2016_nation_table['num_2016_nation']
num_2017_nation_table['num']=num_2017_nation_table['num_2017_nation']
num_2018_nation_table['num']=num_2018_nation_table['num_2018_nation']
num_2019_nation_table['num']=num_2019_nation_table['num_2019_nation']
num_2020_nation_table['num']=num_2020_nation_table['num_2020_nation']
num_2021_nation_table['num']=num_2021_nation_table['num_2021_nation']

nation_df = pd.concat([num_2015_nation_table,num_2016_nation_table,num_2017_nat
nation_df['Year'] = ["2015","2016","2017","2018","2019","2020","2021"]
nation_df = nation_df[['num',"Year"]]
nation_df
```

Out[19]:

	num	Year
0	5803476	2015
0	6237057	2016
0	6660284	2017
0	7000076	2018
0	7459404	2019
0	7942631	2020
0	8530032	2021

In [20]:
```
fig, axs = plt.subplots(1,figsize=(8,8))
fig.suptitle('National Deployment: 2015-2021')

axs.bar(nation_df['Year'], nation_df['num'])
axs.set_ylabel("Num of Blocks")
axs.set_xlabel("Year")
```

Out[20]:
```
Text(0.5, 0, 'Year')
```



State level data: deployment rate change

In [21]:
```
%%bigquery state_deployed_block_nums
SELECT a.StateAbbr, num_2018_state, num_2019_state,num_2020_state,num_2021_stat
FROM

(SELECT StateAbbr, count(distinct BlockCode) as num_2018_state
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2018_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
group by StateAbbr) as a
join (
SELECT StateAbbr, count(distinct BlockCode) as num_2019_state
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2019_v2`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
group by StateAbbr
) as b on a.StateAbbr = b.StateAbbr
join (
SELECT StateAbbr, count(distinct BlockCode) as num_2020_state
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2020_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
group by StateAbbr
) as c on a.StateAbbr = c.StateAbbr
join (
SELECT StateAbbr, count(distinct BlockCode) as num_2021_state
FROM `broadband-data.fcc_form_477.fbd_us_without_satellite_jun2021_v1`
where consumer = 1 and StateAbbr !='AK' and StateAbbr !='HI' and MaxAdDown >= 2
group by StateAbbr
) as d on a.StateAbbr = d.StateAbbr
```

```
Query complete after 0.01s: 100%|██████████| 1/1 [00:00<00:00, 444.45query/
s]
Downloading: 100%|██████████| 49/49 [00:01<00:00, 36.19rows/
s]
```

In [22]:
```
state_deployed_block_nums.head()
```

Out[22]:

	StateAbbr	num_2018_state	num_2019_state	num_2020_state	num_2021_state
0	FL	361602	359196	363689	365987
1	ME	43178	43871	44650	54995
2	MI	218731	238297	239500	254826
3	WI	175305	177929	183234	199157
4	OR	100959	104488	116025	123100

In [23]:
```
%%bigquery states_block_level_num
SELECT State_Abbr,Number_of_2010_Census_Blocks
FROM `broadband-data.fcc_form_477.State_Block_Counts_including_Abbr`
```

```
Query complete after 0.00s: 100%|██████████| 1/1 [00:00<00:00, 391.26query/
s]
Downloading: 100%|██████████| 51/51 [00:01<00:00, 34.57rows/
s]
```

In [24]:
```
states_block_level_num.head()
```

Out[24]:

	State_Abbr	Number_of_2010_Census_Blocks
0	NY	350169
1	FL	484481
2	MN	259777
3	VT	32580
4	OR	196621

In [25]:
```
state_deployed_block_nums.head()
```

Out[25]:

	StateAbbr	num_2018_state	num_2019_state	num_2020_state	num_2021_state
0	FL	361602	359196	363689	365987
1	ME	43178	43871	44650	54995
2	MI	218731	238297	239500	254826
3	WI	175305	177929	183234	199157
4	OR	100959	104488	116025	123100

In [26]:
```
df = pd.merge(states_block_level_num, state_deployed_block_nums, how='inner', l
df.head()
```

Out[26]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
0	NY	350169	NY	291911	304160
1	FL	484481	FL	361602	359196
2	MN	259777	MN	185845	195479
3	VT	32580	VT	16902	17257
4	OR	196621	OR	100959	104488

In [27]:
```
# State level deployment rate

df['2018 deployement rate'] = df.num_2018_state/df.Number_of_2010_Census_Blocks
df['2019 deployement rate'] = df.num_2019_state/df.Number_of_2010_Census_Blocks
df['2020 deployement rate'] = df.num_2020_state/df.Number_of_2010_Census_Blocks
df['2021 deployement rate'] = df.num_2021_state/df.Number_of_2010_Census_Blocks
```

In [28]:
```
df.head()
```

Out[28]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
0	NY	350169	NY	291911	304160
1	FL	484481	FL	361602	359196
2	MN	259777	MN	185845	195479
3	VT	32580	VT	16902	17257
4	OR	196621	OR	100959	104488

So state with the least deployment rate is XX and with the most deployment rate is XX. Deployment rate is calculated by the number of census blocks which have broadband deployed in the state devided by total number of census blocks in that state.

Let's see their deployment rates change over recent years.

In [29]:
```
# TODO: Line Plot
print(len(df))
print(df.StateAbbr.nunique())
df.head()
```

```
49
49
```

Out[29]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
0	NY	350169	NY	291911	304160
1	FL	484481	FL	361602	359196
2	MN	259777	MN	185845	195479
3	VT	32580	VT	16902	17257
4	OR	196621	OR	100959	104488

In [30]:
```
df2=df[['StateAbbr',"2018 deployement rate","2019 deployement rate","2020 deplo
df2=df2.T
```

In [31]:
```
new_header = df2.iloc[0]
df2 = df2[1:]
df2.columns = new_header
```

In [32]:
```
df2.head()
#df2['Year']=df2['StateAbbr']
#df2['Year']=['2018','2019','2020','2021']
```

Out[32]:

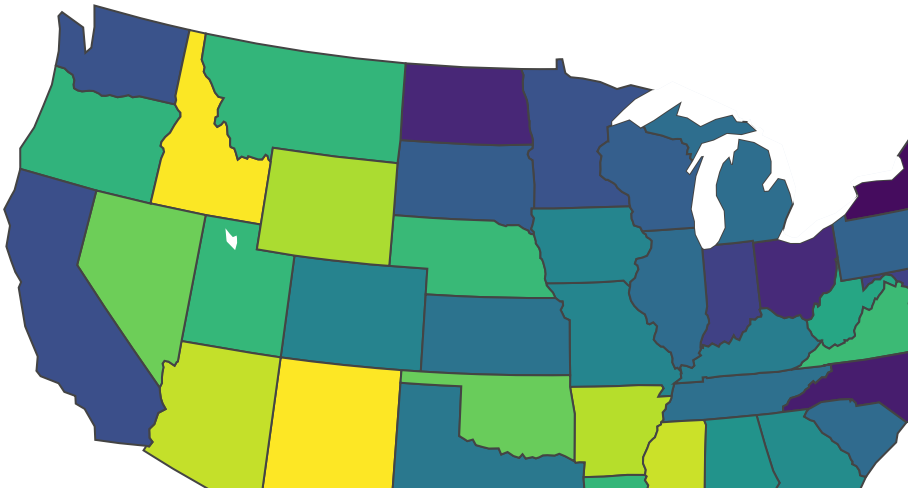
StateAbbr	NY	FL	MN	VT	OR	DE	SD	W
2018 deployement rate	0.833629	0.74637	0.715402	0.518785	0.51347	0.788306	0.696299	0.71511
2019 deployement rate	0.868609	0.741404	0.752488	0.529681	0.531418	0.777732	0.79768	0.72255
2020 deployement rate	0.84116	0.750678	0.839304	0.540884	0.590095	0.766204	0.929697	0.721
2021 deployement rate	0.861064	0.755421	0.885444	0.699448	0.626078	0.793033	0.947929	0.73364

4 rows × 49 columns

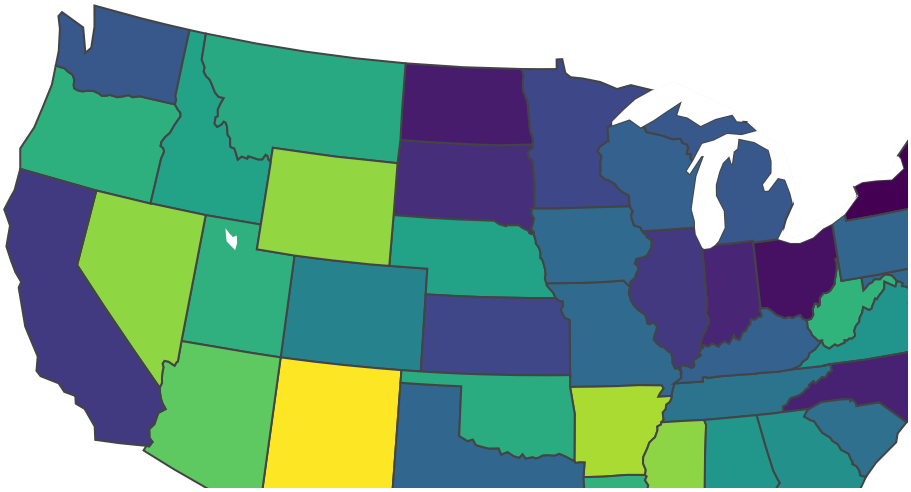
df2.plot(subplots=True, figsize=(40,100)); plt.legend(loc='best')

In [33]:
```
import plotly.express as px
```

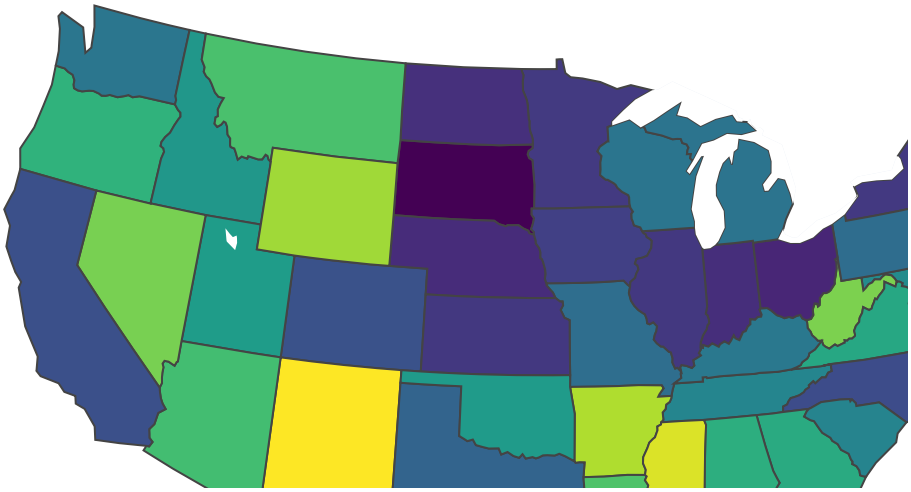
In [34]:
```
#2018
fig = px.choropleth(df,
                    locations='StateAbbr',
                    locationmode="USA-states",
                    scope="usa",
                    color='2018 deployement rate',
                    color_continuous_scale="Viridis_r",

                    )
fig.show()
```



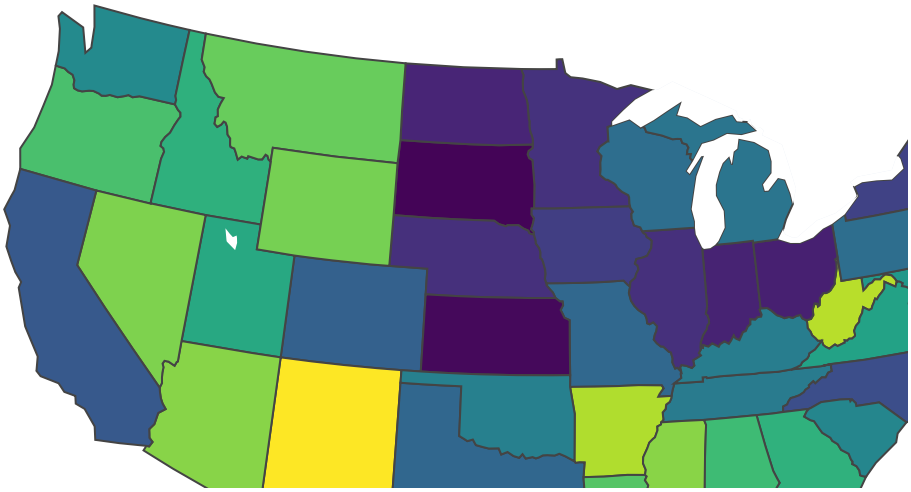
In [35]:
```
#2019
fig = px.choropleth(df,
                    locations='StateAbbr',
                    locationmode="USA-states",
                    scope="usa",
                    color='2019 deployement rate',
                    color_continuous_scale="Viridis_r",

                    )
fig.show()
```



In [36]:
```
#2020
fig = px.choropleth(df,
                    locations='StateAbbr',
                    locationmode="USA-states",
                    scope="usa",
                    color='2020 deployement rate',
                    color_continuous_scale="Viridis_r",

                    )
fig.show()
```



In [37]:
```
#2021
fig = px.choropleth(df,
                    locations='StateAbbr',
                    locationmode="USA-states",
                    scope="usa",
                    color='2021 deployement rate',
                    color_continuous_scale="Viridis_r",

                    )
fig.show()
```



State with the Highest and Lowest Deployment Rate in 2021: Highest Rhode Island; Lowest New Mexico

In [38]:
```
df.loc[df['2021 deployement rate'].idxmax()]
```

Out[38]:
```
State_Abbr                          RI
Number_of_2010_Census_Blocks     25181
StateAbbr                           RI
num_2018_state                   19274
num_2019_state                   19658
num_2020_state                   19696
num_2021_state                   23984
2018 deployement rate         0.765418
2019 deployement rate         0.780668
2020 deployement rate         0.782177
2021 deployement rate         0.952464
Name: 23, dtype: object
```

In [39]:
```
df.loc[df['2021 deployement rate'].idxmin()]
```

Out[39]:
```
State_Abbr                          NM
Number_of_2010_Census_Blocks    168609
StateAbbr                           NM
num_2018_state                   55609
num_2019_state                   56389
num_2020_state                   67790
num_2021_state                   82824
2018 deployement rate          0.32981
2019 deployement rate         0.334436
2020 deployement rate         0.402054
2021 deployement rate         0.491219
Name: 11, dtype: object
```

In [40]:
```
df3=df[['StateAbbr',"2018 deployement rate","2019 deployement rate","2020 deplo
df3.rename(columns={'2018 deployement rate': '2018', '2019 deployement rate': '
                    '2020 deployement rate':'2020','2021 deployement rate':'2021
```

```
/var/folders/f6/cw8ncfy124b8_ngrxb3g5bw40000gn/T/ipykernel_39281/3642516701.p
y:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [41]:
```
df_RI = df3.loc[df3['2021'].idxmax()]
```

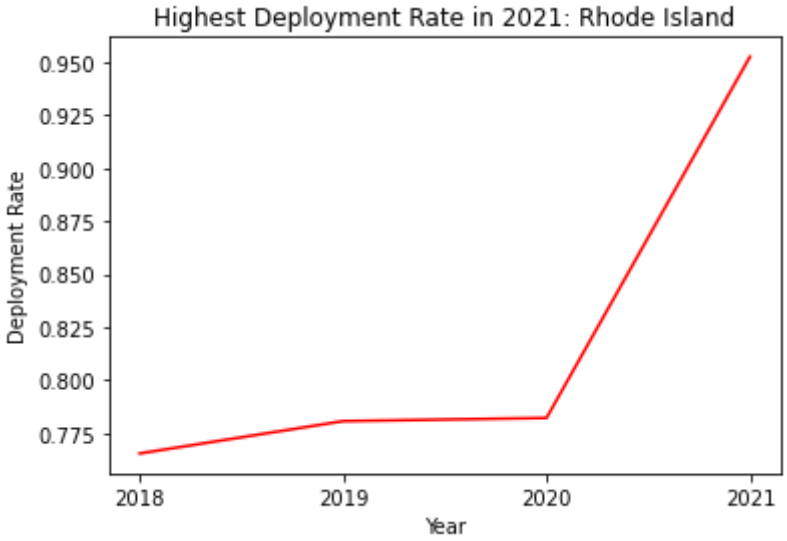
In [42]:
```
new_header = df_RI.iloc[0]
df_RI = df_RI[1:]
df_RI.columns = new_header
```

In [43]:
```
df_RI=df_RI.to_frame()
df_RI
```

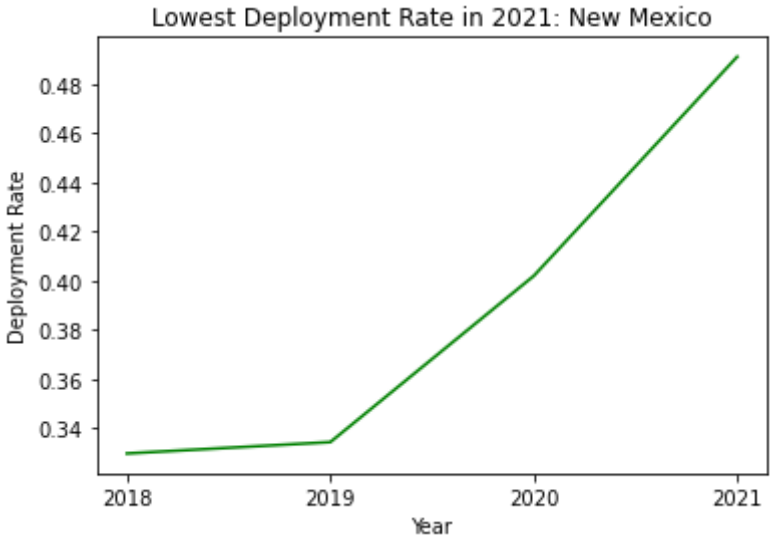
Out[43]:

	23
2018	0.765418
2019	0.780668
2020	0.782177
2021	0.952464

In [44]:
```
plt.plot(df_RI, color='red')
plt.xlabel('Year')
plt.ylabel('Deployment Rate')

plt.title("Highest Deployment Rate in 2021: Rhode Island")

plt.show()
```



In [45]:
```
df_NM = df3.loc[df3['2021'].idxmin()]
new_header = df_NM.iloc[0]
df_NM = df_NM[1:]
df_NM.columns = new_header
df_NM=df_NM.to_frame()
plt.plot(df_NM, color='green')
plt.xlabel('Year')
plt.ylabel('Deployment Rate')

plt.title("Lowest Deployment Rate in 2021: New Mexico")

plt.show()
```



States with the Highest and Lowest Average Deployment Rate

In [46]:
```
df.head()
```

Out[46]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
0	NY	350169	NY	291911	304160
1	FL	484481	FL	361602	359196
2	MN	259777	MN	185845	195479
3	VT	32580	VT	16902	17257
4	OR	196621	OR	100959	104488

In [47]:
```
df['Average Deployment Rate']=df[['2018 deployement rate','2019 deployement rat
                                  '2021 deployement rate']].mean(axis=1)
```

Top 5 States with Highest Deployment Rate: Ohio, New York, North Dakota, South Dakota, Connecticut

In [48]:
```
large5 = df.nlargest(5, "Average Deployment Rate")

large5
```

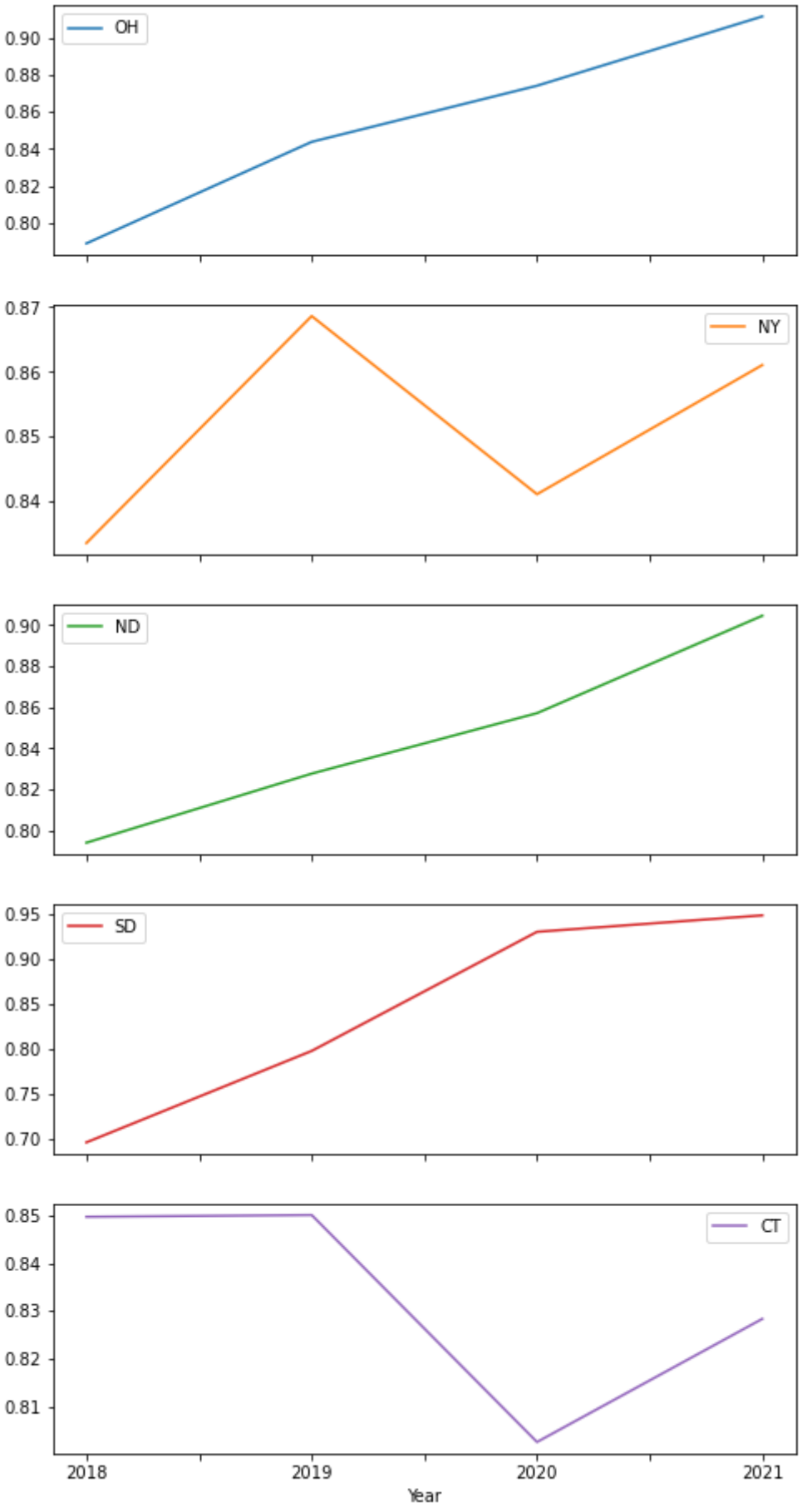
Out[48]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
41	OH	365344	OH	288168	308290
0	NY	350169	NY	291911	304160
21	ND	133769	ND	106214	110715
6	SD	88360	SD	61525	70483
32	CT	67578	CT	57426	57451

In [49]:
```
large5=large5[['StateAbbr',"2018 deployement rate","2019 deployement rate","202
large5.rename(columns={'2018 deployement rate': '2018', '2019 deployement rate'
                       '2020 deployement rate':'2020','2021 deployement rate':'2021
large5=large5.T
new_header = large5.iloc[0]
large5 = large5[1:]
large5.columns = new_header
large5.plot(subplots=True, figsize=(8,16)); plt.legend(loc='best');plt.xlabel('
```

```
/var/folders/f6/cw8ncfy124b8_ngrxb3g5bw40000gn/T/ipykernel_39281/1649292934.p
y:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[49]:
```
Text(0.5, 0, 'Year')
```



Top 5 States with Lowest Average Deployment Rate: New Mexico, Mississippi, Arkansas, Wyoming, Nevada

In [50]:
```
small5 = df.nsmallest(5, "Average Deployment Rate")
small5
```

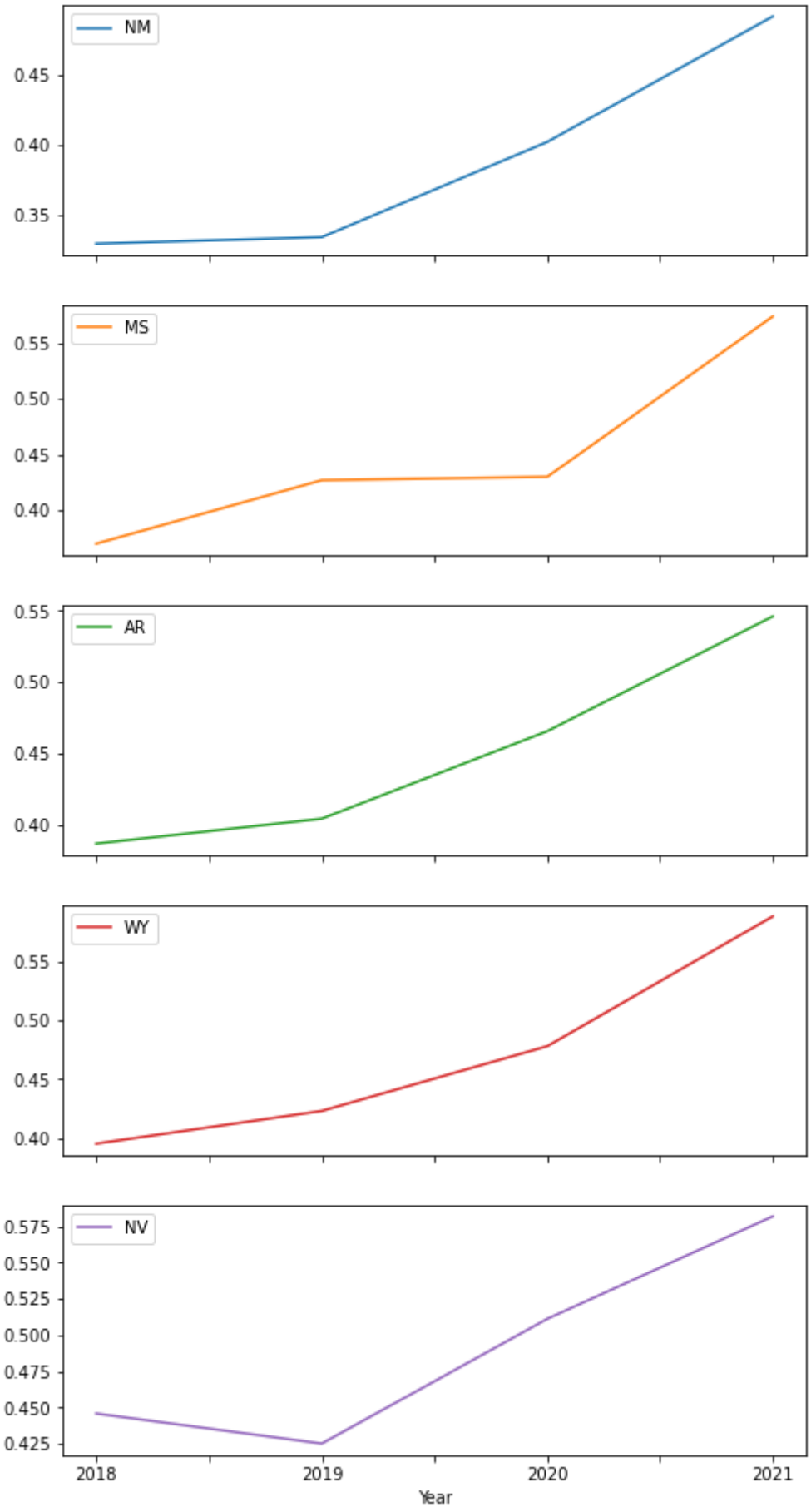
Out[50]:

	State_Abbr	Number_of_2010_Census_Blocks	StateAbbr	num_2018_state	num_2019_state
11	NM	168609	NM	55609	56389
35	MS	171778	MS	63514	73319
13	AR	186211	AR	71977	75236
48	WY	86204	WY	34090	36479
39	NV	84538	NV	37695	35945

In [51]:
```
small5=small5[['StateAbbr',"2018 deployement rate","2019 deployement rate","202
small5.rename(columns={'2018 deployement rate': '2018', '2019 deployement rate'
                   '2020 deployement rate':'2020','2021 deployement rate':'2021
small5=small5.T
new_header = small5.iloc[0]
small5 = small5[1:]
small5.columns = new_header
small5.plot(subplots=True, figsize=(8,16)); plt.legend(loc='best');plt.xlabel('
```

```
/var/folders/f6/cw8ncfy124b8_ngrxb3g5bw40000gn/T/ipykernel_39281/4149072866.p
y:2: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
```

Out[51]:
```
Text(0.5, 0, 'Year')
```


NM
0.45
0.40
0.35
MS
0.55
0.50
0.45
0.40
AR
0.55
0.50
0.45
0.40
WY
0.55
0.50
0.45
0.40
NV
0.575
0.550
0.525
0.500
0.475
0.450
0.425
2018
2019
2020
2021
Year

In [ ]: