Modeling_Capstone_Assignment

April 4, 2022

[45]:
```python
import pandas as pd
import numpy as np
# import geopandas as gpd
from patsy import dmatrices
import statsmodels.api as sm
import os
```

1 CAPSTONE MODELING WRITE UP

1.1 Data Cleaning and Editing

We first include a data cleaning section before we start our modeling portion of the lab

[5]:
```python
# Navigating to csv file
os.chdir('/Users/meera/Documents/drugs/Data')
geogon_od = pd.read_csv('geogon_od.csv')
```

[7]:
```python
geogon_od

# Rename our columns to have no spaces (needed for OLS modeling later)
geogon_od.columns = ['Year', 'FIPS', 'State', 'County', 'Deaths', 'Population',␣
 ↪'Crude_Rate',
       'Cruder_Rate', 'Deathrate_per_100', 'Unemployment_rate',
       'Dispense_rate', 'SUMLEV', 'AGEGRP', 'TOT_POP', 'TOT_MALE',
       'TOT_FEMALE', 'WA_MALE', 'WA_FEMALE', 'BA_MALE', 'BA_FEMALE', 'IA_MALE',
       'IA_FEMALE', 'AA_MALE', 'AA_FEMALE', 'NA_MALE', 'NA_FEMALE', 'TOM_MALE',
       'TOM_FEMALE', 'NH_MALE', 'NH_FEMALE', 'H_MALE', 'H_FEMALE',
       'Urbanicity', 'Jail_Population', 'Incarceration_Rate_per_100k',
       'PovertyCount', 'PovertyPercentage', 'MedianHHI', 'Latitude',
       'Longitude', 'geometry']


# Convert FIPS to string
geogon_od['FIPS'] = geogon_od['FIPS'].astype(str).str.zfill(5)
geogon_od
```

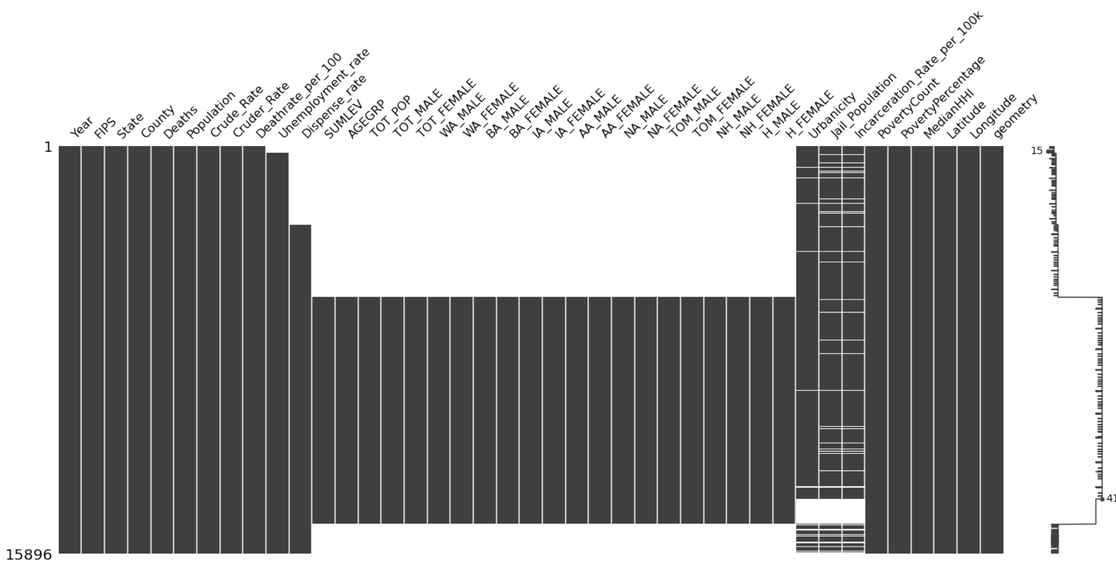
[7]:
```
         Year   FIPS      State                  County  Deaths  Population  \
0      1999.0  01003    Alabama     Baldwin County, AL    11.0    137555.0
1      1999.0  01073    Alabama   Jefferson County, AL    34.0    662845.0
2      1999.0  01089    Alabama     Madison County, AL    10.0    274693.0
3      1999.0  01097    Alabama      Mobile County, AL    28.0    399323.0
4      1999.0  02020     Alaska  Anchorage Borough, AK    21.0    259348.0
…         …     …         …                     …      …          …
15891  2020.0  55139  Wisconsin   Winnebago County, WI    38.0    171631.0
15892  2020.0  55141  Wisconsin        Wood County, WI    18.0     72560.0
15893  2020.0  56021    Wyoming     Laramie County, WY    17.0    100595.0
15894  2020.0  56025    Wyoming     Natrona County, WY    16.0     80815.0
15895  2020.0  56037    Wyoming  Sweetwater County, WY    15.0     42673.0

       Crude_Rate  Cruder_Rate  Deathrate_per_100  Unemployment_rate  …  \
0      Unreliable     7.996801           0.007997                NaN  …
1            5.13     5.129404           0.005129                NaN  …
2      Unreliable     3.640428           0.003640                NaN  …
3            7.01     7.011868           0.007012                NaN  …
4            8.10     8.097228           0.008097                NaN  …
…             …           …                  …                  …  …
15891       22.14    22.140522           0.022141                5.4  …
15892  Unreliable    24.807056           0.024807                6.7  …
15893  Unreliable    16.899448           0.016899                5.1  …
15894  Unreliable    19.798305           0.019798                7.8  …
15895  Unreliable    35.151032           0.035151                7.4  …

       H_FEMALE  Urbanicity  Jail_Population  Incarceration_Rate_per_100k  \
0           NaN   small/mid       390.000000                   440.340000
1           NaN       urban      1779.000000                   408.230000
2           NaN   small/mid       713.000000                   382.130000
3           NaN   small/mid      1223.000000                   470.210000
4           NaN   small/mid              NaN                          NaN
…           …          …               …                            …
15891       NaN         NaN              NaN                          NaN
15892       NaN         NaN              NaN                          NaN
15893       NaN         NaN              NaN                          NaN
15894       NaN         NaN              NaN                          NaN
15895       NaN       rural        83.622951                   197.488525

       PovertyCount  PovertyPercentage  MedianHHI   Latitude   Longitude  \
0           14668.0               10.5    39194.0  30.659218  -87.746067
1           89661.0               13.7    35885.0  33.553444  -86.896536
2           30056.0               11.1    43718.0  34.764238  -86.551080
3           72372.0               18.3    32396.0  30.684572  -88.196568
4           18397.0                7.2    52959.0  61.174250 -149.284329
…               …                 …         …          …          …
15891       14219.0                8.7    64653.0  44.085707  -88.668149
```

```
15892        6732.0                9.4    54154.0  44.461413  -90.038825
15893        7242.0                7.4    69450.0  41.292830 -104.660395
15894        7420.0                9.4    65901.0  42.977645 -106.768219
15895        3187.0                7.6    70583.0  41.660328 -108.875677

                                                  geometry
0      POLYGON ((-88.026319 30.753358, -87.944546 30…
1      POLYGON ((-87.26692299999999 33.512929, -87.27…
2      POLYGON ((-86.78362801716901 34.991924921992, …
3      POLYGON ((-88.432007 31.114297999999998, -88.3…
4      POLYGON ((-150.228774 61.162580999999996, -150…
…                                                    …
15891  POLYGON ((-88.886673 44.242622, -88.7662 44.24…
15892  POLYGON ((-90.31605499999999 44.424502, -90.31…
15893  POLYGON ((-105.27823599999999 41.656655, -104…
15894  POLYGON ((-107.543526 42.781558, -107.501425 4…
15895  POLYGON ((-110.053708 42.270744, -109.49676099…

[15896 rows x 41 columns]
```

1.1.1 Dealing with missing data

[9]:
```python
import missingno as msno
```

[10]:
```python
msno.matrix(geogon_od);
```

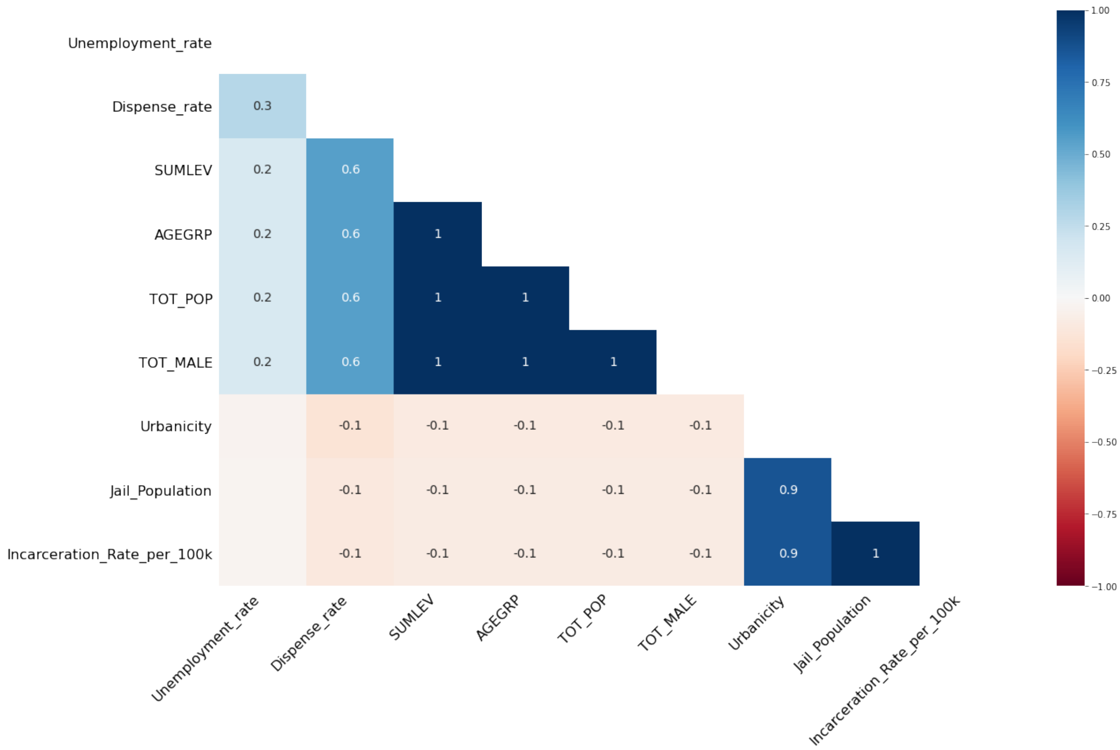


we will only be considering 2010-2019 since we are missing the demographics data from all other years.

we will be adding in the 2018 Incarceration_Rate_per_100k as a replacement for it's 2019 data (note there was a big decrease in incarceration rates in 2020 bc of the quarantine so we did not take an avg of the two adjacent years)

[11]:
```
### Analyze our data without most of the demographic data
geogon_nodem = geogon_od.drop(geogon_od.columns[15:32], axis = 1)
# geogon_od.drop(nonnum_features, axis = 1)
```

[12]:
```
### CORRELATION ACCORDING MISSING VALUES
msno.heatmap(geogon_nodem);
```



The missingno correlation heatmap measures nullity correlation: how strongly the presence or absence of one variable affects the presence of another:

documentation: https://github.com/ResidentMario/missingno

[13]:
```
# detailed numbers for correlations for matrix above
df = geogon_nodem.iloc[:, [i for i, n in enumerate(np.var(geogon_nodem.
↪isnull(), axis='rows')) if n > 0]]
corr_mat = df.isnull().corr()
corr_mat
```

[13]:
```
                             Unemployment_rate  Dispense_rate    SUMLEV  \
Unemployment_rate                     1.000000       0.283027  0.158532
```

```
Dispense_rate                     0.283027       1.000000  0.548387
SUMLEV                            0.158532       0.548387  1.000000
AGEGRP                            0.158532       0.548387  1.000000
TOT_POP                           0.158532       0.548387  1.000000
TOT_MALE                          0.158532       0.548387  1.000000
Urbanicity                       -0.038610      -0.140034 -0.093264
Jail_Population                  -0.025133      -0.108905 -0.079084
Incarceration_Rate_per_100k      -0.025133      -0.108905 -0.079084

                               AGEGRP   TOT_POP  TOT_MALE  Urbanicity  \
Unemployment_rate            0.158532  0.158532  0.158532   -0.038610
Dispense_rate                0.548387  0.548387  0.548387   -0.140034
SUMLEV                       1.000000  1.000000  1.000000   -0.093264
AGEGRP                       1.000000  1.000000  1.000000   -0.093264
TOT_POP                      1.000000  1.000000  1.000000   -0.093264
TOT_MALE                     1.000000  1.000000  1.000000   -0.093264
Urbanicity                  -0.093264 -0.093264 -0.093264    1.000000
Jail_Population             -0.079084 -0.079084 -0.079084    0.866623
Incarceration_Rate_per_100k -0.079084 -0.079084 -0.079084    0.866623

                             Jail_Population  Incarceration_Rate_per_100k
Unemployment_rate                  -0.025133                    -0.025133
Dispense_rate                      -0.108905                    -0.108905
SUMLEV                             -0.079084                    -0.079084
AGEGRP                             -0.079084                    -0.079084
TOT_POP                            -0.079084                    -0.079084
TOT_MALE                           -0.079084                    -0.079084
Urbanicity                          0.866623                     0.866623
Jail_Population                     1.000000                     1.000000
Incarceration_Rate_per_100k         1.000000                     1.000000
```

1.1.2 Filling in missing incarceration data

[14]:
```
geogon_od.groupby('Year').mean()
```

[14]:
```
           Deaths     Population  Cruder_Rate  Deathrate_per_100  \
Year
1999.0  43.555921  572798.555921     8.030704           0.008031
2000.0  41.021407  550509.048930     8.089048           0.008089
2001.0  39.828418  502528.361930     9.292622           0.009293
2002.0  42.926606  454856.417431    11.062642           0.011063
2003.0  40.972441  408695.933071    12.437209           0.012437
2004.0  40.095841  384420.708861    13.196182           0.013196
2005.0  41.963979  373678.807890    13.884325           0.013884
2006.0  43.388805  347913.352496    15.756374           0.015756
2007.0  43.421583  340280.787050    16.184708           0.016185
2008.0  41.411290  325557.427419    16.664888           0.016665
```

```
2009.0  43.087744  334208.793872    16.329976           0.016330
2010.0  42.068123  320799.798201    18.048078           0.018048
2011.0  44.151741  316892.825871    18.898084           0.018898
2012.0  44.847118  321512.358396    18.319839           0.018320
2013.0  46.781327  320574.276413    18.337875           0.018338
2014.0  48.343458  310182.310748    20.294289           0.020294
2015.0  52.496614  305658.584650    21.693002           0.021693
2016.0  60.439583  288510.078125    25.372938           0.025373
2017.0  65.819203  286529.480082    27.561521           0.027562
2018.0  63.169734  288160.408998    25.691622           0.025692
2019.0  66.097760  288708.191446    26.191722           0.026192
2020.0  74.406385  252816.041415    34.250980           0.034251

        Unemployment_rate  Dispense_rate  SUMLEV  AGEGRP        TOT_POP  \
Year
1999.0                NaN            NaN     NaN     NaN            NaN
2000.0           4.004281            NaN     NaN     NaN            NaN
2001.0           4.743164            NaN     NaN     NaN            NaN
2002.0           5.726376            NaN     NaN     NaN            NaN
2003.0           5.991339            NaN     NaN     NaN            NaN
2004.0           5.647378            NaN     NaN     NaN            NaN
2005.0           5.298100            NaN     NaN     NaN            NaN
2006.0           4.874848      92.066415     NaN     NaN            NaN
2007.0           4.742878      95.457698     NaN     NaN            NaN
2008.0           5.892608      98.283737     NaN     NaN            NaN
2009.0           9.419777     100.278273     NaN     NaN            NaN
2010.0           9.869794     104.834961    50.0     0.0  321490.146530
2011.0           9.005348     104.139677    50.0     0.0  316889.021144
2012.0           8.206140     104.281932    50.0     0.0  321460.547619
2013.0           7.476167      99.049754    50.0     0.0  320513.732187
2014.0           6.309813      96.348598    50.0     0.0  309643.820093
2015.0           5.539955      89.287472    50.0     0.0  304896.603837
2016.0           5.145208      82.443646    50.0     0.0  288392.431250
2017.0           4.577630      74.023926    50.0     0.0  285821.787538
2018.0           4.106442      64.474029    50.0     0.0  287726.742331
2019.0           3.856619      51.671894    50.0     0.0  288708.191446
2020.0           7.568939      47.867041     NaN     NaN            NaN

             TOT_MALE  …      NH_FEMALE        H_MALE      H_FEMALE  \
Year                   …
1999.0            NaN  …            NaN           NaN           NaN
2000.0            NaN  …            NaN           NaN           NaN
2001.0            NaN  …            NaN           NaN           NaN
2002.0            NaN  …            NaN           NaN           NaN
2003.0            NaN  …            NaN           NaN           NaN
2004.0            NaN  …            NaN           NaN           NaN
2005.0            NaN  …            NaN           NaN           NaN
```

2006.0	NaN	…	NaN	NaN	NaN
2007.0	NaN	…	NaN	NaN	NaN
2008.0	NaN	…	NaN	NaN	NaN
2009.0	NaN	…	NaN	NaN	NaN
2010.0	157530.796915	…	134361.757069	30201.724936	29597.592545
2011.0	155321.712687	…	132103.347015	29999.366915	29463.961443
2012.0	157589.755639	…	133707.426065	30634.000000	30163.365915
2013.0	157183.609337	…	133217.178133	30577.452088	30112.944717
2014.0	151926.268692	…	128451.280374	29689.603972	29266.271028
2015.0	149642.135440	…	126040.413093	29610.538375	29214.055305
2016.0	141597.475000	…	119205.703125	27962.139583	27589.253125
2017.0	140342.588355	…	117826.941777	28013.385087	27652.257406
2018.0	141273.833333	…	118253.246421	28545.939673	28199.662577
2019.0	141776.857434	…	118379.902240	28890.668024	28551.431772
2020.0	NaN	…	NaN	NaN	NaN

	Jail_Population	Incarceration_Rate_per_100k	PovertyCount \
Year			
1999.0	1365.668772	367.767509	66356.690789
2000.0	1303.648084	362.731136	59734.932722
2001.0	1162.730682	366.685426	56260.646113
2002.0	1074.480865	379.830673	53662.217890
2003.0	990.888971	388.894506	50288.624016
2004.0	946.835695	403.350902	48465.262206
2005.0	929.928615	407.062611	46934.948542
2006.0	881.250457	421.629811	43936.285930
2007.0	855.930089	418.580790	41735.007194
2008.0	827.767061	429.785139	40920.166667
2009.0	830.403977	422.876268	45878.770195
2010.0	767.057795	419.542895	47325.304627
2011.0	737.996795	422.798623	48551.472637
2012.0	736.832477	410.926291	49210.100251
2013.0	734.767103	416.478818	48585.015971
2014.0	708.285349	427.999699	45981.371495
2015.0	657.872240	410.124516	43066.436795
2016.0	622.652948	427.207397	38615.004167
2017.0	618.901575	436.766892	36519.264556
2018.0	614.105153	437.621122	35950.124744
2019.0	NaN	NaN	33712.712831
2020.0	519.032040	265.384844	28892.382226

	PovertyPercentage	MedianHHI	Latitude	Longitude
Year				
1999.0	11.136842	43173.726974	37.915685	-92.371327
2000.0	10.470336	45309.226300	37.931566	-91.039656
2001.0	11.067024	44232.455764	37.669659	-89.731386
2002.0	11.449083	44207.417431	37.834022	-90.865661

```
2003.0          12.100394  44088.791339  37.604391 -90.601783
2004.0          12.589873  44764.520796  37.690760 -90.378170
2005.0          13.316981  46298.166381  37.692520 -90.566926
2006.0          13.477156  47908.633888  37.783387 -89.958696
2007.0          13.179137  49954.489209  37.633187 -90.221341
2008.0          13.444624  51371.474462  37.776175 -90.257119
2009.0          14.503621  49544.293872  37.784857 -90.791144
2010.0          15.663496  48726.704370  37.664978 -90.173991
2011.0          16.031095  49586.343284  37.875412 -89.844187
2012.0          15.994236  50658.586466  37.854009 -89.648570
2013.0          15.774816  52052.468059  37.872434 -89.605800
2014.0          15.473014  53049.012850  38.012852 -89.362848
2015.0          14.889278  54490.230248  38.024084 -89.307257
2016.0          14.224063  56060.640625  38.122730 -88.777908
2017.0          13.712462  58037.790603  38.068656 -88.468890
2018.0          13.512781  59776.467280  38.007860 -88.536241
2019.0          12.656925  63442.706721  38.076585 -88.818858
2020.0          12.609405  63318.083693  38.038889 -88.536673

[22 rows x 34 columns]
```

[16]:
```python
# Get 2019 and 2018 incarceration columns
incar19 = geogon_od[geogon_od['Year'] == 2019][['FIPS',␣
↪'Incarceration_Rate_per_100k']]
incar18 = geogon_od[geogon_od['Year'] == 2018][['FIPS',␣
↪'Incarceration_Rate_per_100k']]

# Merge 2018 rates to 2019 counties
incar19 = incar19.merge(incar18, on = ['FIPS'], how = 'left')
incar19 = incar19.drop('Incarceration_Rate_per_100k_x', axis = 1)

# Place 2019's new rate back into df
geogon_od.loc[geogon_od['Year'] == 2019, 'Incarceration_Rate_per_100k'] = np.
↪array(incar19['Incarceration_Rate_per_100k_y'])
geogon_od[geogon_od['Year'] == 2019]['Incarceration_Rate_per_100k']
```

[16]:
```
13755    477.31
13756    369.20
13757    673.96
13758    676.19
13759    523.14
          …
14732    272.00
14733       NaN
14734       NaN
14735       NaN
14736    516.48
```

```
Name: Incarceration_Rate_per_100k, Length: 982, dtype: float64
```

[17]:
```
# Check the proportion of missing variable that are now present in 2019␣
↪incarcertation rates
geogon_od[geogon_od['Year'] == 2018]['Incarceration_Rate_per_100k'].isnull().
↪sum()
```

[17]:
```
33
```

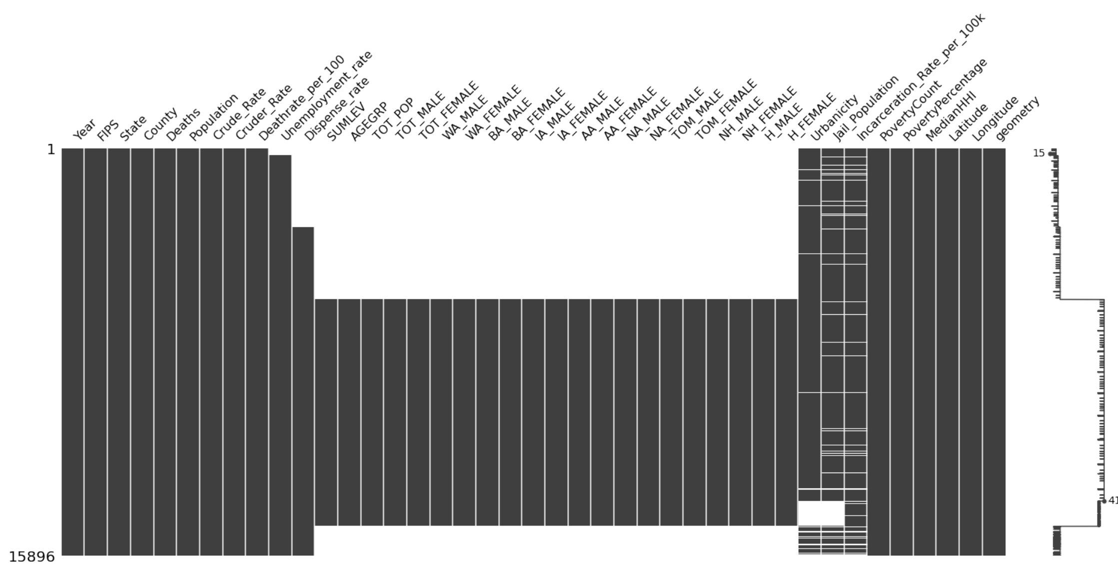
[18]:
```
geogon_od[geogon_od['Year'] == 2019]['Incarceration_Rate_per_100k'].isnull().
↪sum()
```

[18]:
```
143
```

[19]:
```
len(geogon_od[geogon_od['Year'] == 2019]['Incarceration_Rate_per_100k'])
```

[19]:
```
982
```

[21]:
```
msno.matrix(geogon_od);
```



1.1.3 Quantifying the urbanicity column

[22]:
```
geogon_od['Urbanicity'].unique()
```

[22]:
```
array(['small/mid', 'urban', 'rural', 'suburban', nan], dtype=object)
```

[23]:
```
urban_dict = {'rural' : 1, 'small/mid' : 2, 'suburban': 3, 'urban' : 4}
geogon_od = geogon_od.replace({"Urbanicity": urban_dict})
```

```
geogon_od['Urbanicity']
```

[23]:
```
0        2.0
1        4.0
2        2.0
3        2.0
4        2.0
        …
15891    NaN
15892    NaN
15893    NaN
15894    NaN
15895    1.0
Name: Urbanicity, Length: 15896, dtype: float64
```

1.2 Exhaustive Feature Selection

We first use this variable selection technique before we fit our OLS model so that we can see which predictors we should include.

[26]:
```
def standardize(raw_data):
    return ((raw_data - np.mean(raw_data, axis = 0)) / np.std(raw_data, axis =␣
 ↪0))
```

[27]:
```
# Filter to years 2010 - 2019
geogon_dec = geogon_od[(geogon_od['Year'] >= 2010) & (geogon_od['Year'] <=␣
 ↪2019)]
geogon_dec = geogon_dec.dropna()
geogon_decy = geogon_dec.copy()

# Dropping variables that we don't plan to include as covariates
# Excluding female variables
geogon_dec = geogon_dec.drop(['Deaths', 'Deathrate per 100', 'Crude Rate',␣
 ↪'Cruder Rate',
                              'FIPS', 'State', 'County', 'geometry', 'SUMLEV',␣
 ↪'AGEGRP',
                              'Longitude', 'Latitude', 'TOT_POP', 'TOT_FEMALE',␣
 ↪'WA_FEMALE',
                              'BA_FEMALE', 'IA_FEMALE', 'AA_FEMALE',␣
 ↪'NA_FEMALE', 'NH_FEMALE',
                              'TOM_FEMALE', 'H_FEMALE'], axis = 1)
geogon_dec
```

[27]:
```
          Year  Population  Unemployment_rate  Dispense_rate  TOT_MALE  \
5902    2010.0    182265.0                9.9          143.8   89620.0
5903    2010.0     57322.0                9.7           60.1   28385.0
5904    2010.0    118572.0               11.2          182.5   57096.0
```

5905	2010.0	43643.0	10.1	114.7	21603.0	
5906	2010.0	38319.0	11.6	145.7	19784.0	
…	…	…	…	…	…	
13750	2018.0	103718.0	3.0	45.4	51691.0	
13751	2018.0	135693.0	2.6	48.2	67275.0	
13752	2018.0	403072.0	2.7	51.1	197976.0	
13753	2018.0	171020.0	2.8	44.6	85991.0	
13754	2018.0	79115.0	4.6	64.6	39876.0	

	WA_MALE	BA_MALE	IA_MALE	AA_MALE	NA_MALE	TOM_MALE	NH_MALE	\
5902	78717.0	8422.0	661.0	551.0	73.0	1196.0	85166.0	
5903	27415.0	424.0	178.0	48.0	35.0	285.0	25801.0	
5904	44125.0	11327.0	323.0	375.0	63.0	883.0	54969.0	
5905	19011.0	2147.0	109.0	64.0	56.0	216.0	19623.0	
5906	12173.0	6618.0	644.0	38.0	13.0	298.0	19331.0	
…	…	…	…	…	…	…	…	
13750	49477.0	677.0	275.0	535.0	28.0	699.0	45581.0	
13751	64370.0	871.0	255.0	889.0	31.0	859.0	65042.0	
13752	183308.0	3632.0	640.0	7362.0	104.0	2930.0	188228.0	
13753	78554.0	2644.0	701.0	2454.0	45.0	1593.0	82301.0	
13754	37460.0	555.0	661.0	284.0	39.0	877.0	36305.0	

	H_MALE	Urbanicity	Jail Population	Incarceration Rate per 100k	\
5902	4454.0	2.0	734.54	624.90	
5903	2584.0	3.0	124.25	332.87	
5904	2127.0	2.0	505.38	639.24	
5905	1980.0	3.0	175.00	611.12	
5906	453.0	1.0	185.12	730.49	
…	…	…	…	…	
13750	6110.0	1.0	297.00	436.26	
13751	2233.0	3.0	229.00	262.63	
13752	9748.0	3.0	532.00	206.18	
13753	3690.0	2.0	310.00	272.00	
13754	3571.0	2.0	263.00	516.48	

	PovertyCount	PovertyPercentage	MedianHHI
5902	24056.0	13.3	47618.0
5903	9358.0	16.5	42906.0
5904	27152.0	23.5	37916.0
5905	8813.0	20.4	38553.0
5906	9135.0	26.1	31365.0
…	…	…	…
13750	10114.0	10.1	64234.0
13751	6059.0	4.5	75799.0
13752	19937.0	5.0	87333.0
13753	16915.0	10.4	57785.0
13754	7677.0	9.9	64714.0

```
[7591 rows x 19 columns]
```

[28]:
```
geogon_dec.columns
```

[28]:
```
Index(['Year', 'Population', 'Unemployment_rate', 'Dispense_rate', 'TOT_MALE',
       'WA_MALE', 'BA_MALE', 'IA_MALE', 'AA_MALE', 'NA_MALE', 'TOM_MALE',
       'NH_MALE', 'H_MALE', 'Urbanicity', 'Jail Population',
       'Incarceration Rate per 100k', 'PovertyCount', 'PovertyPercentage',
       'MedianHHI'],
      dtype='object')
```

[70]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
from mlxtend.feature_selection import ExhaustiveFeatureSelector as EFS


# Run exhaustive search with linear regression

y = np.array(geogon_decy['Cruder Rate'])

X = np.array(standardize(geogon_dec))

lr = LinearRegression()

efs = EFS(lr,
          min_features=5,
          max_features=19,
          scoring='neg_mean_squared_error',
          cv=5,
          print_progress = True,
          n_jobs = 2)

efs.fit(X, y)

print('Best MSE score: %.2f' % efs.best_score_ * (-1))
print('Best subset:', efs.best_idx_)
```

```
Features: 518507/519252


Best subset: (0, 2, 3, 8, 10, 11, 14, 15, 16, 18)
```

[79]:
```
# Print the variables that EFS determined is the best subset
geogon_dec.columns[[0, 2, 3, 8, 10, 11, 14, 15, 16, 18]]
```

```
[79]: Index(['Year', 'Unemployment_rate', 'Dispense_rate', 'AA_MALE', 'TOM_MALE',
       'NH_MALE', 'Jail Population', 'Incarceration Rate per 100k',
       'PovertyCount', 'MedianHHI'],
      dtype='object')
```

1.3 Calculating VIF

1.3.1 Manually chosen variables

```
[31]: from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
C:\Users\ashmj\anaconda3\lib\site-packages\statsmodels\compat\pandas.py:61:
FutureWarning: pandas.Int64Index is deprecated and will be removed from pandas
in a future version. Use pandas.Index with the appropriate dtype instead.
  from pandas import Int64Index as NumericIndex
```

```
[14]: # Determining which variables to include when calculating VIF

# Excluding string variables and ones that have high VIF

noninclude_features = ['Year', 'FIPS', 'State', 'County', 'Latitude',
                       'Longitude', 'geometry', 'Crude Rate', 'TOT_POP',␣
 ↪'AGEGRP', 'Jail Population',
                       'Deathrate per 100', 'SUMLEV', 'TOT_MALE', 'BA_MALE',␣
 ↪'IA_MALE',
                       'AA_MALE', 'NA_MALE', 'TOM_MALE', 'NH_MALE', 'H_MALE',␣
 ↪'TOT_FEMALE',
                       'WA_FEMALE', 'BA_FEMALE', 'IA_FEMALE', 'AA_FEMALE',␣
 ↪'NA_FEMALE', 'TOM_FEMALE',
                       'NH_FEMALE', 'H_FEMALE', 'Population', 'PovertyCount',␣
 ↪'Deaths', 'MedianHHI', 'Cruder Rate']
geogon_num = geogon_od.drop(noninclude_features, axis = 1)
geogon_num = geogon_num.dropna()
```

```
[16]: # VIF dataframe
vif_data = pd.DataFrame()
vif_data["feature"] = geogon_num.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(geogon_num.values, i)
                          for i in range(len(geogon_num.columns))]

print(vif_data)
```

```
                      feature        VIF
0           Unemployment_rate   8.926970
1               Dispense_rate   8.362615
```

```
2                      WA_MALE   1.726278
3                   Urbanicity   4.999564
4   Incarceration Rate per 100k   4.045233
5             PovertyPercentage  11.713832
```

1.3.2 Best Subset Variables

[36]:
```python
geogon_best = geogon_dec[['Year', 'Unemployment_rate', 'Dispense_rate',␣
↪'AA_MALE', 'TOM_MALE',
       'NH_MALE', 'Jail Population', 'Incarceration Rate per 100k',
       'PovertyCount', 'MedianHHI']]

# VIF dataframe
vif_data = pd.DataFrame()
vif_data["feature"] = geogon_best.columns

# calculating VIF for each feature
vif_data["VIF"] = [variance_inflation_factor(geogon_best.values, i)
                          for i in range(len(geogon_best.columns))]

vif_data = vif_data.set_index('feature')
vif_data
```

[36]:

feature	VIF
Year	68.784791
Unemployment_rate	8.954940
Dispense_rate	10.116277
AA_MALE	5.854949
TOM_MALE	14.211463
NH_MALE	14.546340
Jail Population	9.974360
Incarceration Rate per 100k	4.811989
PovertyCount	13.589621
MedianHHI	31.760316

[40]:
```python
fivenum_best = geogon_best.describe().transpose()
pd.concat([fivenum_best, vif_data], axis = 1)
```

[40]:

	count	mean	std	min \
Year	7591.0	2014.218285	2.585592	2010.00
Unemployment_rate	7591.0	6.542590	2.725402	2.00
Dispense_rate	7591.0	90.727335	40.130801	9.90
AA_MALE	7591.0	8407.628903	34112.541448	5.00
TOM_MALE	7591.0	3730.123040	8132.088403	36.00
NH_MALE	7591.0	117908.745752	179201.669693	2956.00
Jail Population	7591.0	683.893442	1178.096923	3.00

Incarceration Rate per 100k	7591.0	423.926483	278.985128	4.25
PovertyCount	7591.0	42216.501910	91349.717261	1527.00
MedianHHI	7591.0	53625.945330	14747.719159	22289.00

	25%	50%	75%	max \
Year	2012.000	2014.00	2016.000	2018.00
Unemployment_rate	4.500	6.00	8.000	29.40
Dispense_rate	64.000	83.90	109.300	426.40
AA_MALE	296.000	1008.00	4469.000	720458.00
TOM_MALE	699.000	1474.00	3502.000	154085.00
NH_MALE	33490.500	61169.00	127541.500	2539478.00
Jail Population	185.000	347.00	728.500	19091.94
Incarceration Rate per 100k	251.035	358.88	522.285	4265.42
PovertyCount	10491.500	18623.00	40665.500	1873522.00
MedianHHI	43426.000	50612.00	60041.500	140382.00

	VIF
Year	68.784791
Unemployment_rate	8.954940
Dispense_rate	10.116277
AA_MALE	5.854949
TOM_MALE	14.211463
NH_MALE	14.546340
Jail Population	9.974360
Incarceration Rate per 100k	4.811989
PovertyCount	13.589621
MedianHHI	31.760316

Looking at all the VIF scores, there is high multicollinearity within the chosen subset (VIF > 4). The `XX_MALE` columns are likely correlated with each other, since they are all a proportion of the greater population. It seems like `Year` has the highest multicollinearity with other variables because each variable's pattern changes significantly depending on the year. `Jail Population` and `Incarceration Rate per 100k` would also be correlated since they are both variables describing the jail population. `Poverty Count`, `MedianHHI`, and `Unemployment_rate` would also be correlated since they are indicators of county wealth. Some options to decrease the multicollinearity would be to drop certain columns or conduct PCA/LASSO/Ridge Regression.

1.4 OLS Modeling

[25]:
```
def standardize(raw_data):
    return ((raw_data - np.mean(raw_data, axis = 0)) / np.std(raw_data, axis =␣
 ↪0))
```

[46]:
```
## STANDARDIZE OUR NON NUMERIC VARIABLES ##
nonnum_features = ['Year', 'FIPS', 'State', 'County', 'Latitude',
                   'Longitude', 'geometry']
stzd_geogon = standardize(geogon_od.drop(nonnum_features, axis = 1))
```

```
stzd_geogon[nonnum_features] = geogon_od[nonnum_features]
stzd_geogon


# Convert to geopandas
# stzd_geogon = gpd.GeoDataFrame(stzd_geogon)

# Convert FIPS to string
stzd_geogon['FIPS'] = stzd_geogon['FIPS'].astype(str).str.zfill(5)
stzd_geogon.dtypes
```

```
/Users/meera/opt/anaconda3/lib/python3.8/site-
packages/numpy/core/fromnumeric.py:3472: FutureWarning: Dropping of nuisance
columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a
future version this will raise TypeError.  Select only valid columns before
calling the reduction.
  return mean(axis=axis, dtype=dtype, out=out, **kwargs)
/Users/meera/opt/anaconda3/lib/python3.8/site-
packages/numpy/core/fromnumeric.py:3613: FutureWarning: Dropping of nuisance
columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a
future version this will raise TypeError.  Select only valid columns before
calling the reduction.
  return std(axis=axis, dtype=dtype, out=out, ddof=ddof, **kwargs)
```

[46]:
```
AA_FEMALE                      float64
AA_MALE                        float64
AGEGRP                         float64
BA_FEMALE                      float64
BA_MALE                        float64
Crude_Rate                      object
Cruder_Rate                    float64
Deathrate_per_100              float64
Deaths                         float64
Dispense_rate                  float64
H_FEMALE                       float64
H_MALE                         float64
IA_FEMALE                      float64
IA_MALE                        float64
Incarceration_Rate_per_100k    float64
Jail_Population                float64
MedianHHI                      float64
NA_FEMALE                      float64
NA_MALE                        float64
NH_FEMALE                      float64
NH_MALE                        float64
Population                     float64
PovertyCount                   float64
```

```
PovertyPercentage            float64
SUMLEV                       float64
TOM_FEMALE                   float64
TOM_MALE                     float64
TOT_FEMALE                   float64
TOT_MALE                     float64
TOT_POP                      float64
Unemployment_rate            float64
Urbanicity                   float64
WA_FEMALE                    float64
WA_MALE                      float64
Year                         float64
FIPS                          object
State                         object
County                        object
Latitude                     float64
Longitude                    float64
geometry                      object
dtype: object
```

[27]:

```
### FILTER TO WRITE-UP YEARS ###
stzd_geogon15 = stzd_geogon[(stzd_geogon['Year'] >= 2010) &␣
 ↪(stzd_geogon['Year'] <= 2019)]

# rename cols to have no spaces
stzd_geogon15 = stzd_geogon15[['Year', 'AA_FEMALE', 'AA_MALE', 'BA_FEMALE',␣
 ↪'BA_MALE', 'Urbanicity', 'Cruder_Rate', 'Dispense_rate',
                               'H_FEMALE', 'H_MALE', 'IA_FEMALE', 'IA_MALE',␣
 ↪'Incarceration_Rate_per_100k', 'MedianHHI',
                               'NA_FEMALE', 'NA_MALE', 'NH_FEMALE', 'NH_MALE',␣
 ↪'PovertyPercentage', 'TOM_FEMALE',
                               'TOM_MALE', 'TOT_FEMALE', 'TOT_MALE',␣
 ↪'Unemployment_rate', 'WA_FEMALE', 'WA_MALE',
                               'Jail_Population', 'PovertyCount']]
stzd_geogon15
```

[27]:

```
         Year  AA_FEMALE   AA_MALE  BA_FEMALE   BA_MALE  Urbanicity  \
5902   2010.0  -0.231379 -0.238974  -0.235517 -0.240625   -0.304556
5903   2010.0  -0.250251 -0.253012  -0.387928 -0.407600    0.856906
5904   2010.0  -0.239150 -0.243886  -0.159893 -0.179977   -0.304556
5905   2010.0  -0.250201 -0.252565  -0.355643 -0.371629    0.856906
5906   2010.0  -0.250806 -0.253291  -0.294024 -0.278287   -1.466018
…        …         …         …          …         …           …
14732  2019.0  -0.183113 -0.184189  -0.367967 -0.358100         NaN
14733  2019.0  -0.233271 -0.235039  -0.389216 -0.408456         NaN
14734  2019.0  -0.249343 -0.252007  -0.393204 -0.413362         NaN
14735  2019.0  -0.234608 -0.236239  -0.377178 -0.384761         NaN
```

```
14736  2019.0  -0.242329 -0.245951  -0.384744 -0.404406         NaN

       Cruder_Rate  Dispense_rate  H_FEMALE    H_MALE  …  PovertyPercentage  \
5902     -0.409017       1.396442 -0.222770 -0.220213  …          -0.098955
5903     -0.182721      -0.577984 -0.236235 -0.236765  …           0.505728
5904     -0.163833       2.309349 -0.238921 -0.240810  …           1.828472
5905      0.369377       0.709993 -0.241827 -0.242111  …           1.242686
5906      0.432862       1.441262 -0.252166 -0.255627  …           2.319777
…              …             …         …         …  …                …
14732    -0.596206      -0.931824 -0.222127 -0.225966  …          -0.779223
14733    -0.254354      -0.799723 -0.244645 -0.249068  …          -0.590260
14734     0.569543      -0.863414 -0.243360 -0.246590  …          -0.174540
14735    -0.279840      -0.733673 -0.191039 -0.192128  …          -0.817016
14736    -0.265716      -0.245374 -0.225685 -0.227002  …          -0.741431

       TOM_FEMALE  TOM_MALE  TOT_FEMALE  TOT_MALE  Unemployment_rate  \
5902    -0.310467 -0.311222   -0.212214 -0.213458           1.449969
5903    -0.412779 -0.412393   -0.436746 -0.434394           1.372003
5904    -0.345331 -0.345982   -0.324234 -0.330805           1.956744
5905    -0.425703 -0.420056   -0.460908 -0.458864           1.527934
5906    -0.420381 -0.410949   -0.473053 -0.465427           2.112675
…             …         …           …         …                 …
14732   -0.284400 -0.263024   -0.240108 -0.224975          -1.239840
14733   -0.394641 -0.389849   -0.408699 -0.407096          -0.966961
14734   -0.383562 -0.377522   -0.469676 -0.465622          -0.655099
14735   -0.292220 -0.275795   -0.366790 -0.354917          -1.044926
14736   -0.341204 -0.343095   -0.399791 -0.391582          -0.850013

       WA_FEMALE   WA_MALE  Jail_Population  PovertyCount
5902   -0.170838 -0.174712        -0.039441     -0.210906
5903   -0.437599 -0.431517        -0.488617     -0.370065
5904   -0.346355 -0.347871        -0.208104     -0.177380
5905   -0.480043 -0.473585        -0.451265     -0.375966
5906   -0.517263 -0.507815        -0.443816     -0.372479
…            …         …                …             …
14732  -0.184171 -0.174667              NaN     -0.298520
14733  -0.402340 -0.398449              NaN     -0.387650
14734  -0.505498 -0.495510              NaN     -0.417938
14735  -0.351729 -0.338030              NaN     -0.371840
14736  -0.392531 -0.379668              NaN     -0.387639

[8835 rows x 28 columns]
```

[28]:
```
print("sm.OLS will drop", len(stzd_geogon15) - len(stzd_geogon15.dropna()),␣
↪"observations")
stzd_geogon15.dropna()
```

```
sm.OLS will drop 1244 observations
```

[28]:
```
         Year  AA_FEMALE   AA_MALE  BA_FEMALE   BA_MALE  Urbanicity  \
5902   2010.0  -0.231379 -0.238974  -0.235517 -0.240625   -0.304556
5903   2010.0  -0.250251 -0.253012  -0.387928 -0.407600    0.856906
5904   2010.0  -0.239150 -0.243886  -0.159893 -0.179977   -0.304556
5905   2010.0  -0.250201 -0.252565  -0.355643 -0.371629    0.856906
5906   2010.0  -0.250806 -0.253291  -0.294024 -0.278287   -1.466018
…         …         …         …         …         …          …
13750  2018.0  -0.237485 -0.239420  -0.384637 -0.402318   -1.466018
13751  2018.0  -0.226711 -0.229541  -0.378734 -0.398268    0.856906
13752  2018.0  -0.053859 -0.048887  -0.337059 -0.340626    0.856906
13753  2018.0  -0.184678 -0.185863  -0.370882 -0.361253   -0.304556
13754  2018.0  -0.243161 -0.246425  -0.385889 -0.404865   -0.304556

       Cruder_Rate  Dispense_rate  H_FEMALE    H_MALE  …  PovertyPercentage  \
5902     -0.409017       1.396442 -0.222770 -0.220213  …          -0.098955
5903     -0.182721      -0.577984 -0.236235 -0.236765  …           0.505728
5904     -0.163833       2.309349 -0.238921 -0.240810  …           1.828472
5905      0.369377       0.709993 -0.241827 -0.242111  …           1.242686
5906      0.432862       1.441262 -0.252166 -0.255627  …           2.319777
…              …              …         …         …  …                …
13750     0.016646      -0.924747 -0.206874 -0.205555  …          -0.703638
13751    -0.322838      -0.858697 -0.236262 -0.239872  …          -1.761834
13752    -0.153014      -0.790288 -0.170256 -0.173354  …          -1.667352
13753    -0.259066      -0.943618 -0.223192 -0.226976  …          -0.646949
13754    -0.254838      -0.471832 -0.226116 -0.228029  …          -0.741431

       TOM_FEMALE  TOM_MALE  TOT_FEMALE  TOT_MALE  Unemployment_rate  \
5902    -0.310467 -0.311222   -0.212214 -0.213458           1.449969
5903    -0.412779 -0.412393   -0.436746 -0.434394           1.372003
5904    -0.345331 -0.345982   -0.324234 -0.330805           1.956744
5905    -0.425703 -0.420056   -0.460908 -0.458864           1.527934
5906    -0.420381 -0.410949   -0.473053 -0.465427           2.112675
…             …         …           …         …                 …
13750   -0.373353 -0.366416   -0.357141 -0.350306          -1.239840
13751   -0.357930 -0.348647   -0.300361 -0.294079          -1.395771
13752   -0.126154 -0.118653    0.174371  0.177490          -1.356788
13753   -0.288093 -0.267133   -0.242169 -0.226552          -1.317806
13754   -0.345548 -0.346648   -0.400863 -0.392935          -0.616116

       WA_FEMALE   WA_MALE  Jail_Population  PovertyCount
5902   -0.170838 -0.174712        -0.039441     -0.210906
5903   -0.437599 -0.431517        -0.488617     -0.370065
5904   -0.346355 -0.347871        -0.208104     -0.177380
5905   -0.480043 -0.473585        -0.451265     -0.375966
5906   -0.517263 -0.507815        -0.443816     -0.372479
```

```
…          …         …                …           …
13750  -0.329619 -0.321080        -0.361472     -0.361878
13751  -0.253419 -0.246530        -0.411521     -0.405788
13752   0.364868  0.348843        -0.188511     -0.255509
13753  -0.185704 -0.175528        -0.351904     -0.288233
13754  -0.393265 -0.381234        -0.386496     -0.388267

[7591 rows x 28 columns]
```

[29]:
```
"Cruder_Rate ~ AA_FEMALE + AA_MALE + BA_FEMALE + BA_MALE + \
H_FEMALE + H_MALE + IA_FEMALE + IA_MALE + NA_FEMALE + NA_MALE + NH_FEMALE +␣
 ↪NH_MALE + \
TOM_FEMALE + TOM_MALE + WA_FEMALE + WA_MALE + \
Unemployment_rate + Dispense_rate + Incarceration_Rate_per_100k +␣
 ↪PovertyPercentage + MedianHHI"
```

[29]:
```
'Cruder_Rate ~ AA_FEMALE + AA_MALE + BA_FEMALE + BA_MALE + H_FEMALE + H_MALE +
IA_FEMALE + IA_MALE + NA_FEMALE + NA_MALE + NH_FEMALE + NH_MALE + TOM_FEMALE +
TOM_MALE + WA_FEMALE + WA_MALE + Unemployment_rate + Dispense_rate +
Incarceration_Rate_per_100k + PovertyPercentage + MedianHHI'
```

1.4.1 Modeling without spatial covariates

[34]:
```
### MANUALLY CHOSEN FEATURES ###
y, X = dmatrices("Cruder_Rate ~ Year + WA_MALE + Urbanicity + Unemployment_rate␣
 ↪+ \
Dispense_rate + Incarceration_Rate_per_100k + PovertyPercentage + MedianHHI",
                 data=stzd_geogon15, return_type='dataframe')
mod = sm.OLS(y, X)
res = mod.fit()
residuals = res.resid
predicted = res.fittedvalues
observed = y
print(res.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            Cruder_Rate   R-squared:                       0.240
Model:                            OLS   Adj. R-squared:                  0.240
Method:                 Least Squares   F-statistic:                     299.8
Date:                Sun, 03 Apr 2022   Prob (F-statistic):               0.00
Time:                        19:44:52   Log-Likelihood:                -9572.3
No. Observations:                7591   AIC:                         1.916e+04
Df Residuals:                    7582   BIC:                         1.923e+04
Df Model:                           8
Covariance Type:            nonrobust
================================================================================
```

```
===============
                                  coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                    -360.5609     11.472    -31.430      0.000
-383.049    -338.073
Year                            0.1790      0.006     31.439      0.000
0.168       0.190
WA_MALE                        -0.0450      0.011     -4.009      0.000
-0.067      -0.023
Urbanicity                     -0.0230      0.013     -1.815      0.069
-0.048       0.002
Unemployment_rate               0.1494      0.015     10.241      0.000
0.121       0.178
Dispense_rate                   0.3134      0.014     23.191      0.000
0.287       0.340
Incarceration_Rate_per_100k     0.0088      0.011      0.830      0.406
-0.012       0.029
PovertyPercentage              -0.0275      0.018     -1.572      0.116
-0.062       0.007
MedianHHI                      -0.1330      0.020     -6.772      0.000
-0.171      -0.094
==============================================================================
Omnibus:                     3914.240   Durbin-Watson:                   1.362
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            41611.534
Skew:                           2.240   Prob(JB):                         0.00
Kurtosis:                      13.559   Cond. No.                     2.36e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.36e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[33]:

```python
### OLS WITH EXHAUSTIVE BEST SUBSET
y, X = dmatrices("Cruder_Rate ~ Year + Unemployment_rate + Dispense_rate + \
                 AA_MALE + TOM_MALE + NH_MALE + Jail_Population + \
                 Incarceration_Rate_per_100k + PovertyCount + MedianHHI",
                 data=stzd_geogon15, return_type='dataframe')
mod = sm.OLS(y, X)
res = mod.fit()
residuals = res.resid
predicted = res.fittedvalues
observed = y
print(res.summary())
```

```
bss_coef = round(res.params, 3)
bss_pval = round(res.pvalues, 3)
bss_df = pd.concat([bss_coef, bss_pval], axis = 1)
bss_df = bss_df.rename(columns={0: 'Estimate', 1: 'P-value'})
bss_df.loc['spatmax'] = ['N/A', 'N/A']
bss_df.loc['spatmean'] = ['N/A', 'N/A']
bss_df
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            Cruder_Rate   R-squared:                       0.247
Model:                            OLS   Adj. R-squared:                  0.246
Method:                 Least Squares   F-statistic:                     249.0
Date:                Sun, 03 Apr 2022   Prob (F-statistic):               0.00
Time:                        19:44:35   Log-Likelihood:                -9537.4
No. Observations:                7591   AIC:                         1.910e+04
Df Residuals:                    7580   BIC:                         1.917e+04
Df Model:                          10
Covariance Type:            nonrobust
================================================================================
===============
                                 coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                   -367.9896     11.136    -33.045      0.000
-389.819    -346.160
Year                           0.1827      0.006     33.057      0.000
0.172       0.194
Unemployment_rate              0.1521      0.014     10.784      0.000
0.124       0.180
Dispense_rate                  0.2985      0.014     21.930      0.000
0.272       0.325
AA_MALE                        0.0982      0.024      4.073      0.000
0.051       0.145
TOM_MALE                      -0.1459      0.037     -3.920      0.000
-0.219      -0.073
NH_MALE                        0.2223      0.032      7.024      0.000
0.160       0.284
Jail_Population               -0.0997      0.031     -3.231      0.001
-0.160      -0.039
Incarceration_Rate_per_100k    0.0231      0.011      2.017      0.044
0.001       0.046
PovertyCount                  -0.1429      0.033     -4.313      0.000
-0.208      -0.078
MedianHHI                     -0.1742      0.015    -11.574      0.000
```

```
-0.204      -0.145
==============================================================================
Omnibus:                     3866.583   Durbin-Watson:                   1.362
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            40325.914
Skew:                           2.211   Prob(JB):                         0.00
Kurtosis:                      13.390   Cond. No.                     2.30e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.3e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[33]:

	Estimate	P-value
Intercept	-367.99	0.0
Year	0.183	0.0
Unemployment_rate	0.152	0.0
Dispense_rate	0.299	0.0
AA_MALE	0.098	0.0
TOM_MALE	-0.146	0.0
NH_MALE	0.222	0.0
Jail_Population	-0.1	0.001
Incarceration_Rate_per_100k	0.023	0.044
PovertyCount	-0.143	0.0
MedianHHI	-0.174	0.0
spatmax	N/A	N/A
spatmean	N/A	N/A

Note that we did not include spatial components in our exhaustive search.

There is a low R-squared but it's slightly better than manually chosen

F-test is 0 so we reject the null hypothesis that the model with no covariates is better – i.e. this says our model is better nothing! This applies for all of our models.

All our variables are significant. This may be due to our large number of observations (7591 observations vs 10 features).

We compare the AIC of our different models. This exhaustive search (with no spat comp) has the lowest AIC (an indication of a better model).

1.4.2 Modeling with naive spatial components

We hypothesize that there are spatial autocorrelations and spillover effects between counties. So we will include spatmax (maximum drug overdose death rate of adjacent counties) and spatmean (average drug overdose death rate of adjacent counties) as variables of interest in our model.

[36]:

```python
#### CREATING NAIVE SPATCOMP TABLE ###
geogon_spat = geogon_od[(geogon_od['Year'] >= 2010) & (geogon_od['Year'] <=␣
↪2019)]
geogon_spat = geogon_spat.drop(['Latitude', 'Longitude', 'geometry', 'AGEGRP',␣
↪'SUMLEV'], axis=1)

# Quantify Urbanicity
urban_dict = {'rural' : 1, 'small/mid' : 2, 'suburban': 3, 'urban' : 4}
geogon_spat = geogon_spat.replace({"Urbanicity": urban_dict})

geogon_spat
```

[36]:

```
          Year   FIPS      State               County  Deaths  Population  \
5902    2010.0  01003    Alabama    Baldwin County, AL    26.0    182265.0
5903    2010.0  01009    Alabama     Blount County, AL    10.0     57322.0
5904    2010.0  01015    Alabama    Calhoun County, AL    21.0    118572.0
5905    2010.0  01021    Alabama    Chilton County, AL    11.0     43643.0
5906    2010.0  01053    Alabama   Escambia County, AL    10.0     38319.0
…          …      …          …                    …       …           …
14732   2019.0  55139  Wisconsin  Winnebago County, WI    20.0    171907.0
14733   2019.0  55141  Wisconsin       Wood County, WI    12.0     72999.0
14734   2019.0  56013    Wyoming    Fremont County, WY    11.0     39261.0
14735   2019.0  56021    Wyoming    Laramie County, WY    16.0     99500.0
14736   2019.0  56025    Wyoming    Natrona County, WY    13.0     79858.0

        Crude_Rate  Cruder_Rate  Deathrate_per_100  Unemployment_rate  …  \
5902         14.26    14.264944           0.014265                9.9  …
5903    Unreliable    17.445309           0.017445                9.7  …
5904         17.71    17.710758           0.017711               11.2  …
5905    Unreliable    25.204500           0.025205               10.1  …
5906    Unreliable    26.096714           0.026097               11.6  …
…              …            …                  …                  …  …
14732        11.63    11.634198           0.011634                3.0  …
14733   Unreliable    16.438581           0.016439                3.7  …
14734   Unreliable    28.017626           0.028018                4.5  …
14735   Unreliable    16.080402           0.016080                3.5  …
14736   Unreliable    16.278895           0.016279                4.0  …

        NH_MALE  NH_FEMALE  H_MALE  H_FEMALE  Urbanicity  Jail_Population  \
5902    85166.0    89879.0  4454.0    3613.0         2.0           734.54
5903    25801.0    26907.0  2584.0    2084.0         3.0           124.25
5904    54969.0    59533.0  2127.0    1779.0         2.0           505.38
5905    19623.0    20601.0  1980.0    1449.0         3.0           175.00
5906    19331.0    18286.0   453.0     275.0         1.0           185.12
…            …          …       …         …           …               …
14732   82624.0    81793.0  3804.0    3686.0         NaN              NaN
14733   34757.0    35919.0  1194.0    1129.0         NaN              NaN
```

```
14734  18256.0    18256.0  1474.0    1275.0          NaN              NaN
14735  42786.0    41871.0  7627.0    7216.0          NaN              NaN
14736  36564.0    36325.0  3687.0    3282.0          NaN              NaN

       Incarceration_Rate_per_100k  PovertyCount  PovertyPercentage  MedianHHI
5902                        624.90       24056.0               13.3    47618.0
5903                        332.87        9358.0               16.5    42906.0
5904                        639.24       27152.0               23.5    37916.0
5905                        611.12        8813.0               20.4    38553.0
5906                        730.49        9135.0               26.1    31365.0
…                              …            …                 …        …
14732                       272.00       15965.0                9.7    59643.0
14733                          NaN        7734.0               10.7    57325.0
14734                          NaN        4937.0               12.9    57953.0
14735                          NaN        9194.0                9.5    69613.0
14736                       516.48        7735.0                9.9    66104.0

[8835 rows x 36 columns]
```

[37]:
```
spatcomp = pd.read_csv('spatialcomp.csv').reset_index(drop = True)
spatcomp = spatcomp.drop('Unnamed: 0', axis = 1)
spatcomp.columns = ['Year', 'FIPS', 'spatmax', 'spatmean']
spatcomp['FIPS'] =  spatcomp['FIPS'].astype(str).str.zfill(5)
spatcomp
```

[37]:

	Year	FIPS	spatmax	spatmean
0	2010	01001	25.204500	16.526161
1	2010	01003	26.096714	16.138937
2	2010	01005	NaN	NaN
3	2010	01007	25.204500	15.378684
4	2010	01009	50.728854	22.318995
…	…	…	…	…
35459	2020	72149	NaN	NaN
35460	2020	72151	NaN	NaN
35461	2020	72153	NaN	NaN
35462	2020	78020	NaN	NaN
35463	2020	78030	NaN	NaN

[35464 rows x 4 columns]

[38]:
```
# MERGE
geogon_spatcomp = geogon_spat.merge(spatcomp, on = ['Year', 'FIPS'], how =␣
↪'left')

# STANDARDIZE
nostzd_features = ['Year', 'FIPS', 'State', 'County', 'Crude_Rate',␣
↪'Cruder_Rate']
```

```python
stzd_geospat = standardize(geogon_spatcomp.drop(nostzd_features, axis = 1))

stzd_geospat[nostzd_features] = geogon_spatcomp[nostzd_features]


# # DROP NA ROWS
print("sm.OLS will drop", len(stzd_geospat) - len(stzd_geospat.dropna()), "observations")
stzd_geospat.dropna()
```

```
sm.OLS will drop 1769 observations
```

[38]:

```
        Deaths  Population  Deathrate_per_100  Unemployment_rate  \
0    -0.317829   -0.214403          -0.583092           1.340094
1    -0.497956   -0.435446          -0.354042           1.266964
2    -0.374119   -0.327086          -0.334924           1.815445
3    -0.486698   -0.459646           0.204776           1.413225
4    -0.497956   -0.469065           0.269033           1.961707
…          …          …                 …                …
7847 -0.441666   -0.332598          -0.674772          -1.365746
7848 -0.374119   -0.353365          -0.152249          -1.182919
7849 -0.374119   -0.296796          -0.495863          -1.329181
7850  0.200036    0.176238          -0.323973          -1.292615
7851 -0.295313   -0.234297          -0.431316          -1.256050

      Dispense_rate   TOT_POP  TOT_MALE  TOT_FEMALE   WA_MALE  WA_FEMALE  …  \
0          1.427865 -0.212854 -0.213458   -0.212214 -0.174712  -0.170838  …
1         -0.624997 -0.435650 -0.434394   -0.436746 -0.431517  -0.437599  …
2          2.377038 -0.327505 -0.330805   -0.324234 -0.347871  -0.346355  …
3          0.714146 -0.459966 -0.458864   -0.460908 -0.473585  -0.480043  …
4          1.474466 -0.469371 -0.465427   -0.473053 -0.507815  -0.517263  …
…               …        …        …          …        …        …  …
7847      -1.022325 -0.333131 -0.327395   -0.338576 -0.309652  -0.320166  …
7848      -0.985535 -0.353832 -0.350306   -0.357141 -0.321080  -0.329619  …
7849      -0.916861 -0.297317 -0.294079   -0.300361 -0.246530  -0.253419  …
7850      -0.845734  0.175926  0.177490    0.174371  0.348843   0.364868  …
7851      -1.005156 -0.234532 -0.226552   -0.242169 -0.175528  -0.185704  …

      PovertyPercentage  MedianHHI   spatmax  spatmean    Year   FIPS  \
0             -0.260658  -0.479678 -0.225981 -0.549059  2010.0  01003
1              0.336784  -0.787107  1.159894 -0.017769  2010.0  01009
2              1.643689  -1.112673 -0.684671 -0.671256  2010.0  01015
3              1.064917  -1.071113 -0.944412 -0.790751  2010.0  01021
4              2.129111  -1.540085 -0.839379 -0.832000  2010.0  01053
…                   …         …        …        …       …      …
7847          -1.343523   0.452526 -0.769278 -0.577038  2018.0  55117
7848          -0.858101   0.604413 -0.107726 -0.203059  2018.0  55127
```

```
7849          -1.903625   1.358958  0.418120 -0.043379  2018.0  55131
7850          -1.810274   2.111480  0.418120 -0.086436  2018.0  55133
7851          -0.802090   0.183656 -0.820830 -1.039795  2018.0  55139

          State                County  Crude_Rate  Cruder_Rate
0       Alabama     Baldwin County, AL       14.26    14.264944
1       Alabama      Blount County, AL  Unreliable    17.445309
2       Alabama     Calhoun County, AL       17.71    17.710758
3       Alabama     Chilton County, AL  Unreliable    25.204500
4       Alabama    Escambia County, AL  Unreliable    26.096714
…           …                     …          …           …
7847  Wisconsin   Sheboygan County, WI  Unreliable    12.991962
7848  Wisconsin    Walworth County, WI       20.25    20.247209
7849  Wisconsin  Washington County, WI       15.48    15.476112
7850  Wisconsin    Waukesha County, WI       17.86    17.862814
7851  Wisconsin   Winnebago County, WI       16.37    16.372354

[7066 rows x 38 columns]
```

[39]:

```
# OLS WITH EXHAUSTIVE BEST SUBSET + spatmax
y, X = dmatrices("Cruder_Rate ~ Year + Unemployment_rate + Dispense_rate + \
                 AA_MALE + TOM_MALE + NH_MALE + Jail_Population + \
                 Incarceration_Rate_per_100k + PovertyCount + MedianHHI +␣
 ↪spatmax",
                 data=stzd_geospat, return_type='dataframe')
mod = sm.OLS(y, X)
res = mod.fit()
residuals = res.resid
predicted = res.fittedvalues
observed = y
print(res.summary())


spatmax_coef = round(res.params, 3)
spatmax_pval = round(res.pvalues, 3)
spatmax_df = pd.concat([spatmax_coef, spatmax_pval], axis = 1)
spatmax_df = spatmax_df.rename(columns={0: 'Estimate', 1: 'P-value'})
spatmax_df.loc['spatmean'] = ['N/A', 'N/A']
spatmax_df
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            Cruder_Rate   R-squared:                       0.490
Model:                            OLS   Adj. R-squared:                  0.489
Method:                 Least Squares   F-statistic:                     616.4
Date:                Sun, 03 Apr 2022   Prob (F-statistic):               0.00
Time:                        19:49:46   Log-Likelihood:                -26279.
```

```
No. Observations:                7066   AIC:                         5.258e+04
Df Residuals:                    7054   BIC:                         5.266e+04
Df Model:                          11
Covariance Type:            nonrobust
================================================================================
===============
                                 coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                  -2593.7068    144.006    -18.011      0.000
-2876.003   -2311.411
Year                           1.2987      0.071     18.167      0.000
1.159       1.439
Unemployment_rate              0.9582      0.184      5.195      0.000
0.597       1.320
Dispense_rate                  2.0658      0.164     12.587      0.000
1.744       2.388
AA_MALE                        0.9110      0.284      3.209      0.001
0.354       1.468
TOM_MALE                      -1.1197      0.439     -2.553      0.011
-1.979      -0.260
NH_MALE                        2.0340      0.375      5.423      0.000
1.299       2.769
Jail_Population               -0.7084      0.316     -2.243      0.025
-1.328      -0.089
Incarceration_Rate_per_100k    0.7123      0.145      4.902      0.000
0.427       0.997
PovertyCount                  -1.5026      0.388     -3.868      0.000
-2.264      -0.741
MedianHHI                     -2.5208      0.183    -13.809      0.000
-2.879      -2.163
spatmax                        7.3515      0.130     56.565      0.000
7.097       7.606
==============================================================================
Omnibus:                     2928.886   Durbin-Watson:                   1.883
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            28098.756
Skew:                           1.723   Prob(JB):                         0.00
Kurtosis:                      12.141   Cond. No.                     2.44e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.44e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[39]:

	Estimate	P-value
Intercept	-2593.707	0.0
Year	1.299	0.0
Unemployment_rate	0.958	0.0
Dispense_rate	2.066	0.0
AA_MALE	0.911	0.001
TOM_MALE	-1.12	0.011
NH_MALE	2.034	0.0
Jail_Population	-0.708	0.025
Incarceration_Rate_per_100k	0.712	0.0
PovertyCount	-1.503	0.0
MedianHHI	-2.521	0.0
spatmax	7.351	0.0
spatmean	N/A	N/A

R-squared higher than prev models. ~ doubled!

F-test is 0 so we reject the null hypothesis that the model with no covariates is better – i.e. this says our model is better nothing! This applies for all of our models.

All our variables are significant. This may be due to our large number of observations (8835 observations vs 11 features).

We compare the AIC of our different models. This model has the highest AIC out of all the models.

[41]:

```
# OLS WITH EXHAUSTIVE BEST SUBSET + spatmean
y, X = dmatrices("Cruder_Rate ~ Year + Unemployment_rate + Dispense_rate + \
                 AA_MALE + TOM_MALE + NH_MALE + Jail_Population + \
                 Incarceration_Rate_per_100k + PovertyCount + MedianHHI +␣
 ↪spatmean",
                 data=stzd_geospat, return_type='dataframe')
mod = sm.OLS(y, X)
res = mod.fit()
residuals = res.resid
predicted = res.fittedvalues
observed = y
print(res.summary())


spatmean_coef = round(res.params, 3)
spatmean_pval = round(res.pvalues, 3)
spatmean_df = pd.concat([spatmean_coef, spatmean_pval], axis = 1)
spatmean_df = spatmean_df.rename(columns={0: 'Estimate', 1: 'P-value'})
spatmean_df.loc['spatmax'] = ['N/A', 'N/A']
spatmean_df = spatmean_df.reindex(['Intercept', 'Year', 'Unemployment_rate',␣
 ↪'Dispense_rate', 'AA_MALE',
       'TOM_MALE', 'NH_MALE', 'Jail_Population', 'Incarceration_Rate_per_100k',
       'PovertyCount', 'MedianHHI','spatmax', 'spatmean'])
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            Cruder_Rate   R-squared:                       0.527
Model:                            OLS   Adj. R-squared:                  0.527
Method:                 Least Squares   F-statistic:                     715.3
Date:                Sun, 03 Apr 2022   Prob (F-statistic):               0.00
Time:                        19:50:10   Log-Likelihood:                -26011.
No. Observations:                7066   AIC:                         5.205e+04
Df Residuals:                    7054   BIC:                         5.213e+04
Df Model:                          11
Covariance Type:            nonrobust
================================================================================
===============
                                 coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                  -2080.2567    140.294    -14.828      0.000
-2355.275   -1805.239
Year                           1.0438      0.070     14.989      0.000
0.907       1.180
Unemployment_rate              1.1610      0.177      6.549      0.000
0.813       1.509
Dispense_rate                  1.5605      0.159      9.795      0.000
1.248       1.873
AA_MALE                        0.3874      0.274      1.415      0.157
-0.149       0.924
TOM_MALE                      -0.5324      0.423     -1.259      0.208
-1.361       0.296
NH_MALE                        1.9405      0.361      5.373      0.000
1.233       2.649
Jail_Population               -0.5558      0.304     -1.827      0.068
-1.152       0.040
Incarceration_Rate_per_100k    0.7304      0.140      5.220      0.000
0.456       1.005
PovertyCount                  -1.3567      0.374     -3.627      0.000
-2.090      -0.623
MedianHHI                     -1.8099      0.176    -10.262      0.000
-2.156      -1.464
spatmean                       8.2374      0.130     63.291      0.000
7.982       8.493
==============================================================================
Omnibus:                     2958.626   Durbin-Watson:                   1.964
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            31758.368
Skew:                           1.707   Prob(JB):                         0.00
Kurtosis:                      12.809   Cond. No.                     2.47e+06
==============================================================================
```

```
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.47e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

R-squared increase from spatmax to spatmean

F-test is 0 so we reject the null hypothesis that the model with no covariates is better – i.e. this says our model is better nothing! This applies for all of our models.

All our variables are significant, except AA_MALE and TOM_MALE. This may be because spatmean is able to explain more variation in our model. Most of our variables are significant - this may be due to our large number of observations (8835 observations vs 11 features).

We compare the AIC of our different models. This model has a relatively high AIC.

[42]:
```
# OLS WITH EXHAUSTIVE BEST SUBSET + SPATCOMP
y, X = dmatrices("Cruder_Rate ~ Year + Unemployment_rate + Dispense_rate + \
                 AA_MALE + TOM_MALE + NH_MALE + Jail_Population + \
                 Incarceration_Rate_per_100k + PovertyCount + MedianHHI +␣
↪spatmax + spatmean",
                 data=stzd_geospat, return_type='dataframe')
mod = sm.OLS(y, X)
res = mod.fit()
residuals = res.resid
predicted = res.fittedvalues
observed = y
print(res.summary())


spatcomp_coef = round(res.params, 3)
spatcomp_pval = round(res.pvalues, 3)
spatcomp_df = pd.concat([spatcomp_coef, spatcomp_pval], axis = 1)
spatcomp_df = spatcomp_df.rename(columns={0: 'Estimate', 1: 'P-value'})
spatcomp_df
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            Cruder_Rate   R-squared:                       0.527
Model:                            OLS   Adj. R-squared:                  0.527
Method:                 Least Squares   F-statistic:                     655.8
Date:                Sun, 03 Apr 2022   Prob (F-statistic):               0.00
Time:                        19:50:29   Log-Likelihood:                -26011.
No. Observations:                7066   AIC:                         5.205e+04
Df Residuals:                    7053   BIC:                         5.214e+04
Df Model:                          12
Covariance Type:            nonrobust
================================================================================
===============
```

```
                                 coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                   -2083.6440    140.337    -14.847      0.000
-2358.748   -1808.540
Year                            1.0455      0.070     15.008      0.000
0.909       1.182
Unemployment_rate               1.1482      0.178      6.459      0.000
0.800       1.497
Dispense_rate                   1.5663      0.159      9.824      0.000
1.254       1.879
AA_MALE                         0.4029      0.274      1.469      0.142
-0.135       0.940
TOM_MALE                       -0.5466      0.423     -1.292      0.196
-1.376       0.283
NH_MALE                         1.9376      0.361      5.365      0.000
1.230       2.646
Jail_Population                -0.5593      0.304     -1.839      0.066
-1.156       0.037
Incarceration_Rate_per_100k     0.7311      0.140      5.225      0.000
0.457       1.005
PovertyCount                   -1.3592      0.374     -3.633      0.000
-2.092      -0.626
MedianHHI                      -1.8345      0.178    -10.296      0.000
-2.184      -1.485
spatmax                         0.3153      0.324      0.974      0.330
-0.319       0.950
spatmean                        7.9350      0.337     23.570      0.000
7.275       8.595
==============================================================================
Omnibus:                     2956.358   Durbin-Watson:                   1.964
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            31567.161
Skew:                           1.707   Prob(JB):                         0.00
Kurtosis:                      12.775   Cond. No.                     2.47e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.47e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[42]:

```
                             Estimate  P-value
 Intercept                   -2083.644    0.000
 Year                            1.045    0.000
 Unemployment_rate               1.148    0.000
```

```
Dispense_rate                    1.566    0.000
AA_MALE                          0.403    0.142
TOM_MALE                        -0.547    0.196
NH_MALE                          1.938    0.000
Jail_Population                 -0.559    0.066
Incarceration_Rate_per_100k      0.731    0.000
PovertyCount                    -1.359    0.000
MedianHHI                       -1.834    0.000
spatmax                          0.315    0.330
spatmean                         7.935    0.000
```

R-squared did not increase when we include both spatmax and spatmean.

All our variables are significant, except AA_MALE, TOM_MALE, and spatmax. Therefore, spatmax is not significant when we include spatmean. (spatmean > spatmax). This may be because spatmean is able to explain more variation in our model. Most of our variables are significant - this may be due to our large number of observations (8835 observations vs 11 features).

F-test is 0 so we reject the null hypothesis that the model with no covariates is better – i.e. this says our model is better nothing! This applies for all of our models.

We compare the AIC of our different models. This model has a relatively high AIC (also same AIC as the model with just spatmean).

NOTE:

The features discussed above may not the best measure of our model's performance. In future analysis, we will be plotting our residuals to determine whether or not it is appropriate to use an OLS fit on our data. Additionally, we will also be using cross validation to measure the accuracy of our model.

[43]:
```python
# bss_df.columns = pd.MultiIndex.from_product([['bss_df'], bss_df.columns])
# bss_df
# spatmax_coef = pd.MultiIndex.from_product([['BSS OLS w/ Adjacent County␣
↪Max'], B.columns])
# spatmean.columns = pd.MultiIndex.from_product([['BSS OLS w/ Adjacent County␣
↪Max'], B.columns])
# pd.concat([A, B], axis = 1)
ols_outputs = pd.concat([bss_df, spatmax_df, spatmean_df, spatcomp_df],
          axis = 1, keys=(['Best Subset (BSS) OLS',
                           'BSS OLS w/ spatmax',
                           'BSS OLS w/ spatmean',
                           'BSS OLS w/ spatmax & spatmean']))

# from tabulate import tabulate
# print(tabulate(ols_outputs))
ols_outputs.shape
```

[43]: (13, 8)

[44]:
```
from IPython.display import display, HTML
display(HTML(ols_outputs.to_html()))
```

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

1.4.3 Next Steps

As seen above there is still a lot that must be done in terms of fine tuning our model. We plan to introduce weighted linear regression using queen weights in our future modeling. On a seperate notebook we explored Morran's I and found that with our given data it is important to take into account different geospatial components. We also will normalize our cruder rate and take into account weights for specific counties, which will acount for their specific population size. We also must plot our residuals to see how they behave and we will try to reduce the collinearity between different variables in our model using PCA, Lasso, or Ridge. Also, we will introduce new variables in our model by finding even more data that can describe the population in these counties.