Final_Notebook1_Housing_Transit_and_Census_Data

March 15, 2021

1 Final Notebook 1: Housing Characteristic Analysis and Transit Analysis

1.1 What's in this notebook?

In this notebook, we conducted comparative analysis on the housing affordability between 2014 and 2018. There are two goals of this notebook. First, we want to extract data from housing analysis (housing affordability) and transit analysis (transit density) to conduct correlation analysis with other metrics (which can be found in other notebook). Second, we want to conduct correlation analysis between transit density and migration as these two variables are the most important factors in our analysis, and being able to understand their relationship will be beneficial for the analysis between them and other factors.

1.2 Preliminary findings in this notebook

There are some critical findings from this notebook. We found that all counties in New York Metro Area experienced increases in owner-occupied housing affordability, while some experienced a minor decrease in rental housing affordability. For transit analysis, we found that the counties with the highest transit density are conceentrated in counties close to New York City, and as the distance between a county and NYC increases, its transit density declines. In addition, we discovered the significant positive relationship between transit density and migration (both inflow and outflow).

1.3 Data visualization

For visualization, we created maps and charts to visualize housing affordability, transit lines and stations,and transit density in the region. We also conducted point analysis to deepen our understanding of transit density in New York Metro Area.

2 Preparation

[1]:
```
#import libraries
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import contextily as ctx
from sodapy import Socrata
#for mapping
import folium
```

```
from folium import plugins
from folium.plugins import MarkerCluster
import seaborn as sns
from pointpats import centrography
import numpy
from matplotlib.patches import Ellipse
import plotly.express as px
```

```
/opt/conda/lib/python3.8/site-packages/geopandas/_compat.py:106: UserWarning:
The Shapely GEOS version (3.8.1-CAPI-1.13.3) is incompatible with the GEOS
version PyGEOS was compiled with (3.9.0-CAPI-1.16.2). Conversions between both
will be slow.
  warnings.warn(
```

[2]:
```
#import county boundary
cb=gpd.read_file('border.json')
cb.head()
```

[2]:

	id	admin_name	admin_fips	state	state_fips	name	suffix	pop	sq_miles	prim_miles	countyp010	geometry
0	wg010mf7692.1	Autauga County	01001	AL	01	Autauga	County	54571	604.343394	128.600141	2678	MULTIPOLYGON (((-86.41311 32.70738, -86.41304 …
1	wg010mf7692.2	Barbour County	01005	AL	01	Barbour	County	16589	904.042556	155.323189	2760	MULTIPOLYGON (((-85.25783 32.14792, -85.25849 …
2	wg010mf7692.3	Bibb County	01007	AL	01	Bibb	County	155547	626.168670	118.501658	2599	MULTIPOLYGON (((-87.02586 33.22254, -87.02584 …
3	wg010mf7692.4	Blount County	01009	AL	01	Blount	County	123010	650.579312	148.117852	2435	MULTIPOLYGON (((-86.47602 34.25990, -86.46427 …
4	wg010mf7692.5	Bullock County	01011	AL	01	Bullock	County	10914	625.394547	128.235791	2738	MULTIPOLYGON (((-85.87201 32.27482, -85.87118 …

3 Housing Affordability

[3]:
```
#import 2014 and 2018 housing affordability dataset
hao2014 = pd.read_csv('hao2014.csv')
hao2014 = hao2014.drop([0])
hao2018 = pd.read_csv('hao2018.csv')
hao2018 = hao2018.drop([0])
```

[4]:
```
#add a new column in county border dataset "FIPS"
cb['FIPS']=cb['admin_fips']
#change the datatype of FIPS
cb['FIPS'] = cb['FIPS'].astype(str).astype(int)
```

3.1 Data exploration

[5]:
```
#look at data types
hao2014.dtypes
```

[5]:
```
GEO_ID         object
NAME           object
FIPS           object
DP04_0079E     object
DP04_0080E     object
                ...
DP04_0139E     object
DP04_0139PE    object
DP04_0140E     object
DP04_0140PE    object
DP04_0141E     object
Length: 116, dtype: object
```

Since the housing affordability 2014 and 2018 datasets are from the same source, so I know the data type will be the same across them. In addition, I also conducted data exploration of these data in the first few weeks into the class, I know what I need to do to treat the data before start the analysis.

[6]:
```
#change the 2014 and 2018 housing affordability FIPS data type
hao2014['FIPS'] = hao2014['FIPS'].astype(str).astype(int)
hao2018['FIPS'] = hao2018['FIPS'].astype(str).astype(int)
```

[7]:
```
#rename some columns in hao2018 dataset that I am going to use for analysis to␣
 ↪avoid confusion between two datasets
hao2018.columns=['GEO_ID',
 'NAME',
 'FIPS',
 'DP04_0080E',
 'DP04_0081E',
 'DP04_0081PE',
 'DP04_0082E',
 'DP04_0082PE',
 'DP04_0083E',
 'DP04_0083PE',
 'DP04_0084E',
 'DP04_0084PE',
 'DP04_0085E',
```

```
'DP04_0085PE',
'DP04_0086E',
'DP04_0086PE',
'DP04_0087E',
'DP04_0087PE',
'DP04_0088E',
'DP04_0088PE',
'DP04_0089E',
'DP04_0090PE',
'DP04_0091E',
'DP04_0091PE',
'DP04_0092E',
'DP04_0092PE',
'DP04_0093E',
'DP04_0093PE',
'DP04_0094E',
'DP04_0094PE',
'DP04_0098E',
'DP04_0098PE',
'DP04_0099E',
'DP04_0099PE',
'DP04_0100E',
'DP04_0100PE',
'DP04_0098E.1',
'DP04_0098PE.1',
'DP04_0099E.1',
'DP04_0099PE.1',
'DP04_0100E.1',
'DP04_0100PE.1',
'DP04_0102E',
'DP04_0103E',
'DP04_0103PE',
'DP04_0104E',
'DP04_0104PE',
'DP04_0105E',
'DP04_0105PE',
'DP04_0106E',
'DP04_0106PE',
'DP04_0107E',
'DP04_0107PE',
'DP04_0108E_y',
'DP04_0108PE_y',
'DP04_0109E_y',
'DP04_0110E_y',
'DP04_0110PE_y',
'DP04_0111E_y',
'DP04_0111PE_y',
```

```
'DP04_0112E_y',
'DP04_0112PE_y',
'DP04_0113E_y',
'DP04_0113PE_y',
'DP04_0114E_y',
'DP04_0114PE_y',
'DP04_0115E_y',
'DP04_0115PE_y',
'DP04_0116E_y',
'DP04_0117E_y',
'DP04_0120E_y',
'DP04_0120PE_y',
'DP04_0121E',
'DP04_0121PE',
'DP04_0122E',
'DP04_0122PE',
'DP04_0123E',
'DP04_0123PE',
'DP04_0124E',
'DP04_0124PE',
'DP04_0125E',
'DP04_0126E',
'DP04_0127E',
'DP04_0127PE',
'DP04_0128E',
'DP04_0128PE',
'DP04_0129E',
'DP04_0129PE',
'DP04_0130E',
'DP04_0130PE',
'DP04_0131E',
'DP04_0131PE',
'DP04_0132E',
'DP04_0132PE',
'DP04_0133E',
'DP04_0133PE',
'DP04_0134E',
'DP04_0135E',
'Unnamed: 98',
'DP04_0137E',
'DP04_0137PE',
'DP04_0138E',
'DP04_0138PE',
'DP04_0139E',
'DP04_0139PE',
'DP04_0140E',
'DP04_0140PE',
```

```
 'DP04_0141E',
 'DP04_0141PE',
 'DP04_0142E',
 'DP04_0142PE',
 'DP04_0143E']
```

[8]:
```
#trim couunty border data set according to FIPS code of the counties in NYMA
newcb = cb[cb.FIPS.isin(["34037",
 "36111",
 "36103",
 "34039",
 "36027",
 "36059",
 "34023",
 "36119",
 "09009",
 "34017",
 "42089",
 "36085",
 "36079",
 "34025",
 "34035",
 "34029",
 "09001",
 "09005",
 "34027",
 "34013",
 "36081",
 "34003",
 "36047",
 "36061",
 "34031",
 "36087",
 "34019",
 "42103",
 "36071",
 "36005",
 "34021"])]
newcb.head()
```

[8]:

	id	admin_name	admin_fips	state	state_fips	\
352	wg010mf7692.273	Litchfield County	09005	CT	09	
353	wg010mf7692.274	New Haven County	09009	CT	09	
1692	wg010mf7692.1647	Middlesex County	34023	NJ	34	
1693	wg010mf7692.1648	Monmouth County	34025	NJ	34	
1694	wg010mf7692.1649	Morris County	34027	NJ	34	

```
            name  suffix     pop    sq_miles  prim_miles  countyp010  \
352   Litchfield  County  189927  944.531099  143.390210         742
353    New Haven  County  862477  619.868798  170.575772         824
1692   Middlesex  County   10959  314.088885   99.996252        1060
1693    Monmouth  County  630380  475.639443  173.523904        1088
1694      Morris  County   12934  481.387018  131.986283         944

                                                 geometry   FIPS
352   MULTIPOLYGON (((-73.39166 42.04953, -73.37524 …   9005
353   MULTIPOLYGON (((-72.93470 41.61544, -72.93431 …   9009
1692  MULTIPOLYGON (((-74.29509 40.59449, -74.29389 …  34023
1693  MULTIPOLYGON (((-73.99727 40.47624, -73.99734 …  34025
1694  MULTIPOLYGON (((-74.49702 41.03445, -74.49701 …  34027
```

[9]:
```
#merge housing affordability datasets with county border with FIPS
tracts2014=cb.merge(hao2014,on="FIPS")
tracts2018=cb.merge(hao2018,on="FIPS")
```

[10]:
```
#convert the data type of columns in 2014 housing affordability dataframe from␣
 ↪object to integers for mapping
tracts2014["DP04_0109E"] =tracts2014["DP04_0109E"].astype(str).astype(int)
tracts2014["DP04_0109PE"] =tracts2014["DP04_0109PE"].astype(float).astype(int)
tracts2014["DP04_0110E"] =tracts2014["DP04_0110E"].astype(str).astype(int)
tracts2014["DP04_0110PE"] =tracts2014["DP04_0110PE"].astype(float).astype(int)
tracts2014["DP04_0111E"] =tracts2014["DP04_0111E"].astype(float).astype(int)
tracts2014["DP04_0111PE"] =tracts2014["DP04_0111PE"].astype(float).astype(int)
tracts2014["DP04_0112E"] =tracts2014["DP04_0112E"].astype(float).astype(int)
tracts2014["DP04_0112PE"] =tracts2014["DP04_0112PE"].astype(float).astype(int)
tracts2014["DP04_0113E"] =tracts2014["DP04_0113E"].astype(float).astype(int)
tracts2014["DP04_0113PE"] =tracts2014["DP04_0113PE"].astype(float).astype(int)
tracts2014["DP04_0114E"] =tracts2014["DP04_0114E"].astype(float).astype(int)
```

[11]:
```
#convert the data type of columns in 2018 housing affordability dataframe from␣
 ↪object to integers for mapping
tracts2018["DP04_0111E_y"] =tracts2018["DP04_0111E_y"].astype(str).astype(int)
tracts2018["DP04_0111PE_y"] =tracts2018["DP04_0111PE_y"].astype(float).
 ↪astype(int)
tracts2018["DP04_0112E_y"] =tracts2018["DP04_0112E_y"].astype(str).astype(int)
tracts2018["DP04_0112PE_y"] =tracts2018["DP04_0112PE_y"].astype(float).
 ↪astype(int)
tracts2018["DP04_0113E_y"] =tracts2018["DP04_0113E_y"].astype(float).astype(int)
tracts2018["DP04_0113PE_y"] =tracts2018["DP04_0113PE_y"].astype(float).
 ↪astype(int)
tracts2018["DP04_0114E_y"] =tracts2018["DP04_0114E_y"].astype(float).astype(int)
tracts2018["DP04_0114PE_y"] =tracts2018["DP04_0114PE_y"].astype(float).
 ↪astype(int)
```

```
tracts2018["DP04_0115E_y"] =tracts2018["DP04_0115E_y"].astype(float).astype(int)
tracts2018["DP04_0115PE_y"] =tracts2018["DP04_0115PE_y"].astype(float).
 ↪astype(int)
tracts2018["DP04_0116E_y"] =tracts2018["DP04_0116E_y"].astype(float).astype(int)
```

3.2 Analyze housing affordability

[12]:
```
#calculating the SMOCAPI<30 for 2014 by adding three SMOCAPI categories
tracts2014['SMOCAPI1430']=␣
 ↪tracts2014['DP04_0109PE']+tracts2014['DP04_0110PE']+tracts2014['DP04_0111PE']
tracts2014.head()
```

[12]:

	id	admin_name	admin_fips	state	state_fips	\
0	wg010mf7692.273	Litchfield County	09005	CT	09	
1	wg010mf7692.274	New Haven County	09009	CT	09	
2	wg010mf7692.1647	Middlesex County	34023	NJ	34	
3	wg010mf7692.1648	Monmouth County	34025	NJ	34	
4	wg010mf7692.1649	Morris County	34027	NJ	34	

	name	suffix	pop	sq_miles	prim_miles	…	DP04_0137E	\
0	Litchfield	County	189927	944.531099	143.390210	…	2324	
1	New Haven	County	862477	619.868798	170.575772	…	12362	
2	Middlesex	County	10959	314.088885	99.996252	…	13532	
3	Monmouth	County	630380	475.639443	173.523904	…	6322	
4	Morris	County	12934	481.387018	131.986283	…	5975	

	DP04_0137PE	DP04_0138E	DP04_0138PE	DP04_0139E	DP04_0139PE	DP04_0140E	\
0	15.2	1965	12.9	1604	10.5	5820	
1	11	12281	10.9	10813	9.6	54174	
2	14.4	10280	11	8192	8.7	37366	
3	11.4	6267	11.3	5003	9	27207	
4	14.3	5227	12.5	3514	8.4	15380	

	DP04_0140PE	DP04_0141E	SMOCAPI1430
0	38.1	1813	59
1	48.1	7865	59
2	39.8	4456	56
3	49.1	3602	56
4	36.8	2107	59

[5 rows x 129 columns]

[13]:
```
#calculating the SMOCAPI<30 for 2018 by adding three SMOCAPI categories
tracts2018['SMOCAPI1830']=␣
 ↪tracts2018['DP04_0111PE_y']+tracts2018['DP04_0112PE_y']+tracts2018['DP04_0113PE_y']
tracts2018.head()
```

[13]:
```
                 id         admin_name admin_fips state state_fips  \
0   wg010mf7692.273  Litchfield County      09005    CT         09
1   wg010mf7692.274   New Haven County      09009    CT         09
2  wg010mf7692.1647   Middlesex County      34023    NJ         34
3  wg010mf7692.1648    Monmouth County      34025    NJ         34
4  wg010mf7692.1649      Morris County      34027    NJ         34

          name  suffix     pop    sq_miles  prim_miles  …  DP04_0139E  \
0  Litchfield  County  189927  944.531099  143.390210  …        2207
1   New Haven  County  862477  619.868798  170.575772  …       14309
2   Middlesex  County   10959  314.088885   99.996252  …       13350
3    Monmouth  County  630380  475.639443  173.523904  …        7149
4      Morris  County   12934  481.387018  131.986283  …        6648

  DP04_0139PE  DP04_0140E DP04_0140PE DP04_0141E DP04_0141PE DP04_0142E  \
0        14.1        2089        13.4       1307         8.4       6312
1        12.3       14755        12.6      10224         8.8      52137
2        13.5       11811          12       8422         8.5      39155
3        12.3        6236        10.8       4731         8.2      27317
4          15        5372        12.1       3646         8.2      15316

  DP04_0142PE DP04_0143E SMOCAPI1830
0        40.3       1263          68
1        44.7       8834          65
2        39.7       5603          64
3        47.2       3665          63
4        34.6       1973          66

[5 rows x 125 columns]
```

[14]:
```
#exclude 5 NYC counties from analysis and plotting since they are most likely␣
 ↪to be outliers
tracts2018=tracts2018.drop([24,25,26,27,17])
tracts2018.head()
```

[14]:
```
                 id         admin_name admin_fips state state_fips  \
0   wg010mf7692.273  Litchfield County      09005    CT         09
1   wg010mf7692.274   New Haven County      09009    CT         09
2  wg010mf7692.1647   Middlesex County      34023    NJ         34
3  wg010mf7692.1648    Monmouth County      34025    NJ         34
4  wg010mf7692.1649      Morris County      34027    NJ         34

          name  suffix     pop    sq_miles  prim_miles  …  DP04_0139E  \
0  Litchfield  County  189927  944.531099  143.390210  …        2207
1   New Haven  County  862477  619.868798  170.575772  …       14309
2   Middlesex  County   10959  314.088885   99.996252  …       13350
3    Monmouth  County  630380  475.639443  173.523904  …        7149
```

```
4      Morris  County   12934  481.387018  131.986283  …        6648

  DP04_0139PE  DP04_0140E DP04_0140PE DP04_0141E DP04_0141PE DP04_0142E  \
0        14.1        2089        13.4       1307         8.4       6312
1        12.3       14755        12.6      10224         8.8      52137
2        13.5       11811          12       8422         8.5      39155
3        12.3        6236        10.8       4731         8.2      27317
4          15        5372        12.1       3646         8.2      15316

  DP04_0142PE DP04_0143E SMOCAPI1830
0        40.3       1263          68
1        44.7       8834          65
2        39.7       5603          64
3        47.2       3665          63
4        34.6       1973          66

[5 rows x 125 columns]
```

3.2.1 Analyzing Owner-Occupied Housing Affordability

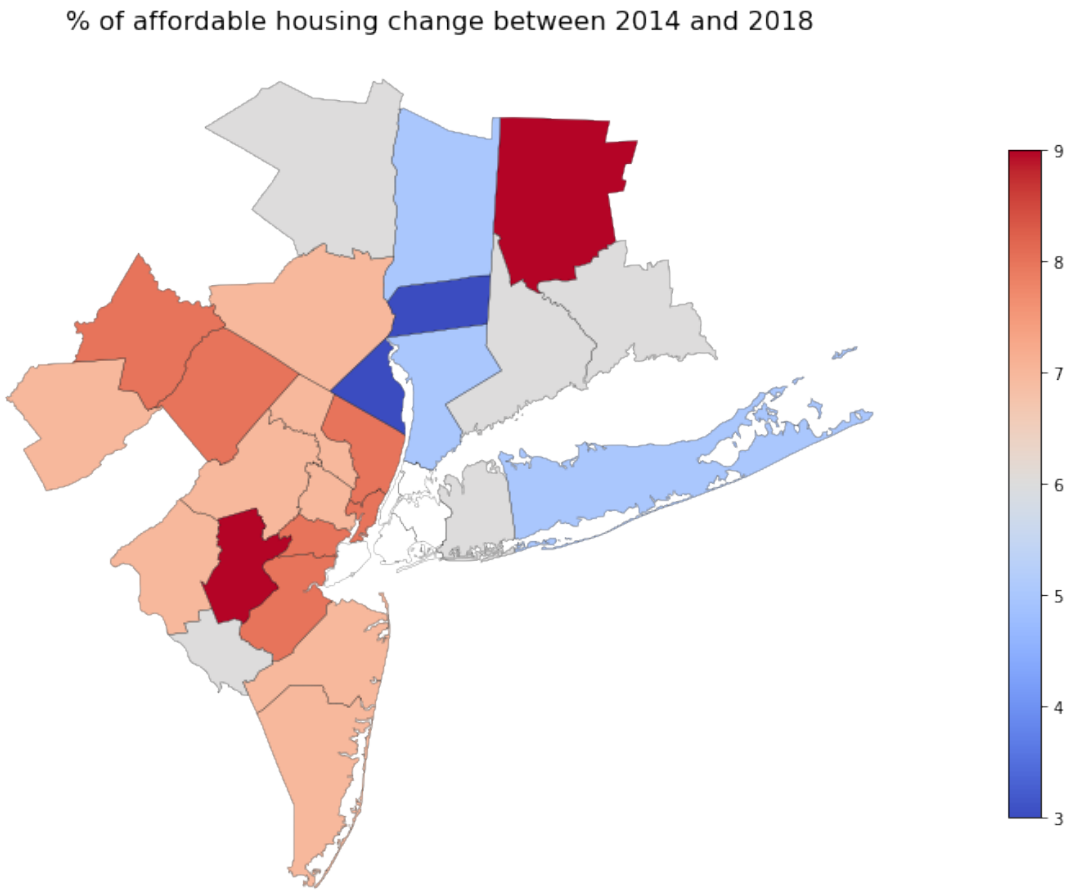
[15]:
```
#calculate housing affordability change betwen 2014 and 2018
tracts2018['ac']=tracts2018['SMOCAPI1830']-tracts2014['SMOCAPI1430']
#get top 5 (oadt5) and bottom 5 (oacb5) counties in NYMA experienced most and␣
 ↪least amount of changes in affordability
oact5=tracts2018 .sort_values(by='ac',ascending = False).head(5)
oacb5=tracts2018 .sort_values(by='ac',ascending = False).tail(5)
#create new dataframe by combining the top 5 and bottom 5 counties for plotting
ooaffordability=oact5.append(oacb5)
```

[16]:
```
#make an interactive charts to show the change of housing affordability between␣
 ↪2014 and 2018 on county level
figac=px.bar(
      ooaffordability,
      x='ac',
      y='NAME',
      orientation='h', #change orientation of bar chart
      color='ac',
      labels={'ac':'% change','NAME_x':'County Names'},#change labels
      color_continuous_scale='Bluered' #change color of the bars
)
#update bar charts
figac.update_layout(title={
        'text': "% of affordable housing change in top five and bottom five␣
 ↪counties", #add title
        'y':1,#change position of the title
        'x':0.5}
```

```
)
figac
```

[17]:
```
#plot the chane of housing affordability on the map
fig, ax = plt.subplots(figsize = (20,10))
tracts2018.plot(
    cmap='coolwarm',
    column='ac',
    legend = True,
    ax=ax,
    legend_kwds={'shrink': 0.75}) #change the size of legend
newcb.geometry.plot(facecolor="none",edgecolor='black',linewidth = 0.2,ax=ax)
plt.axis('off');
plt.title('% of affordable housing change between 2014 and 2018',fontsize=16)
```

[17]: Text(0.5, 1.0, '% of affordable housing change between 2014 and 2018')



3.2.2 Analyzing Rental Housing Affordability

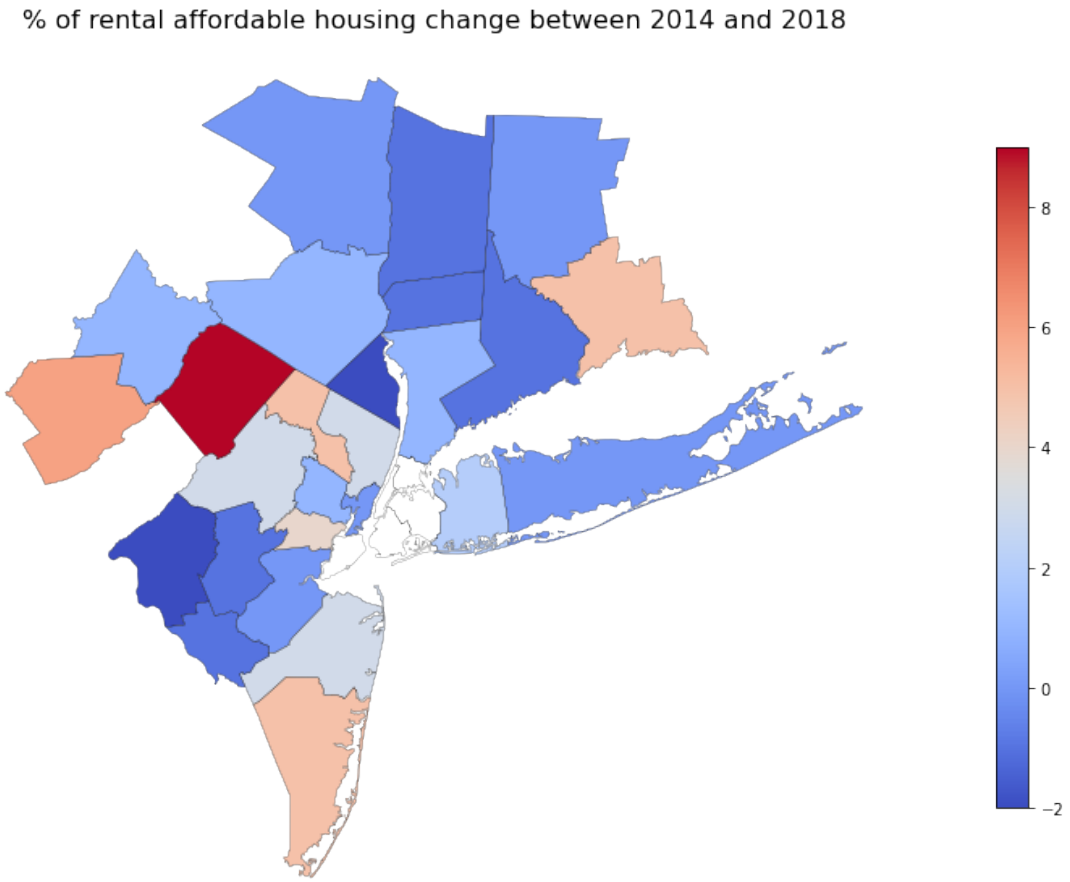
[18]:
```
#changing the datatype for columns with rental housing affordability
tracts2014["DP04_0135PE"] =tracts2014["DP04_0135PE"].astype(float).astype(int)
tracts2014["DP04_0136PE"] =tracts2014["DP04_0136PE"].astype(float).astype(int)
tracts2014["DP04_0137PE"] =tracts2014["DP04_0137PE"].astype(float).astype(int)
tracts2014["DP04_0138PE"] =tracts2014["DP04_0138PE"].astype(float).astype(int)
tracts2018["DP04_0137PE"] =tracts2018["DP04_0137PE"].astype(float).astype(int)
tracts2018["DP04_0138PE"] =tracts2018["DP04_0138PE"].astype(float).astype(int)
tracts2018["DP04_0139PE"] =tracts2018["DP04_0139PE"].astype(float).astype(int)
tracts2018["DP04_0140PE"] =tracts2018["DP04_0140PE"].astype(float).astype(int)
```

[19]:
```
#calculate housing affordability change for rental housing units between 2014␣
 ↪and 2018
tracts2014['GRAPI1430']=␣
 ↪tracts2014['DP04_0135PE']+tracts2014['DP04_0136PE']+tracts2014['DP04_0137PE']+tracts2014['D
tracts2018['GRAPI1830']=␣
 ↪tracts2018['DP04_0137PE']+tracts2018['DP04_0138PE']+tracts2018['DP04_0139PE']+tracts2018['D
tracts2018['rac']=tracts2018['GRAPI1830']-tracts2014['GRAPI1430']
#get top 5 (radt5) and bottom 5 (racb5) counties in NYMA experienced most and␣
 ↪least amount of changes in affordability
ract5=tracts2018 .sort_values(by='rac',ascending = False).head(5)
racb5=tracts2018 .sort_values(by='rac',ascending = False).tail(5)
#create new dataframe by combining the top 5 and bottom 5 counties for plotting
raffordability=ract5.append(racb5)
```

[20]:
```
#look at the change of housing affordability for rental housing between 2014␣
 ↪and 2018 on county level
figac=px.bar(
      raffordability,
      x='rac',
      y='NAME',
      orientation='h', #change orientation of bar chart
      color='rac',
      labels={'rac':'% change','NAME_x':'County Names'},#change labels
      color_continuous_scale='Bluered' #change color of the bars
)
#update bar charts
figac.update_layout(title={
        'text': "% of rental affordable housing change in top five and bottom␣
 ↪five counties", #add title
        'y':1,#change position of the title
        'x':0.5}
)
figac
```

[21]:
```
#plot the chane of rental housing affordability on the map
fig, ax = plt.subplots(figsize = (20,10))
tracts2018.plot(
    cmap='coolwarm',
    column='rac',
    legend = True,
    ax=ax,
    legend_kwds={'shrink': 0.75}) #change the size of legend
newcb.geometry.plot(facecolor="none",edgecolor='black',linewidth = 0.2,ax=ax)
plt.axis('off');
plt.title('% of rental affordable housing change between 2014 and␣
 ↪2018',fontsize=16)
```

[21]:
```
Text(0.5, 1.0, '% of rental affordable housing change between 2014 and 2018')
```



4 Transit Density

4.1 Data Exploration for Transit Lines

[22]:
```
#import transit datasets
njrail=gpd.read_file('NJRail_line/Passenger_Railroad_Lines_in_NJ.shp')
njstation=gpd.read_file('NJRail_station/Railroad_Stations_in_NJ.shx')
lirail=gpd.read_file('nyu-2451-34753-geojson.json')
listation=gpd.read_file('nyu-2451-34754-geojson.json')
mnrail=gpd.read_file('MNStation/mnline.json')
mnstation=gpd.read_file('MNStation/stops.json')
```

[23]:
```
#change the column name for new jersey rail datasets
njrail.columns=['id','linename','service','shape_leng','date_stamp','geometry']
#add a new column indicating that those lines are operated by New Jersey␣
 ↪Railroad
njrail['Operating'] = 'New Jersey Railroad'
njrail.head()
```

[23]:

	id	linename	service	shape_leng	date_stamp	\
0	1	ATLANTIC CITY RAIL LINE	None	356957.019400	2016-08-30	
1	2	BERGEN COUNTY LINE	HOBOKEN	155780.575084	2013-11-04	
2	3	MAIN LINE	HOBOKEN	161721.051881	2013-11-04	
3	4	MEADOWLANDS RAIL LINE	None	56328.562168	2013-11-04	
4	5	MONTCLAIR BOONTON LINE	NEW YORK CITY	328910.733006	2013-11-04	

	geometry	Operating
0	LINESTRING (508669.853 193016.598, 505026.387 …	New Jersey Railroad
1	LINESTRING (622908.638 692949.426, 620720.513 …	New Jersey Railroad
2	LINESTRING (587611.064 830753.567, 588462.132 …	New Jersey Railroad
3	LINESTRING (622908.638 692949.426, 620720.513 …	New Jersey Railroad
4	LINESTRING (399357.761 735253.132, 399513.899 …	New Jersey Railroad

[24]:
```
#repeat the last step to the Long Island Railroad
lirail.columns=['id','number','linename','geometry']
lirail['Operating'] = 'Long Island Railroad'
lirail.head()
```

[24]:

	id	number	linename	\
0	nyu_2451_34753.1	11	Belmont	
1	nyu_2451_34753.2	10	Port Jefferson	
2	nyu_2451_34753.3	12	City Zone	
3	nyu_2451_34753.4	1	Babylon	
4	nyu_2451_34753.5	3	Oyster Bay	

	geometry	Operating
0	MULTILINESTRING ((-73.99309 40.75074, -73.9924…	Long Island Railroad

```
1  MULTILINESTRING ((-73.90300 40.74607, -73.9034…  Long Island Railroad
2  MULTILINESTRING ((-73.80933 40.69955, -73.8100…  Long Island Railroad
3  MULTILINESTRING ((-73.99309 40.75074, -73.9924…  Long Island Railroad
4  MULTILINESTRING ((-73.99309 40.75074, -73.9924…  Long Island Railroad
```

[25]:
```
#convert the geographic coordination system for Long Island Railroad
lirail=lirail.to_crs('epsg:3424')
```

[26]:
```
#Repeat it for Metro-North railroad
mnrail.columns=['id','number','linename','geometry']
mnrail['Operating'] = 'Metro North'
mnrail.head()
```

[26]:
```
                 id  number    linename  \
0  nyu_2451_34755.1       1      Hudson
1  nyu_2451_34755.2       3   New Haven
2  nyu_2451_34755.3       2      Harlem
3  nyu_2451_34755.4       5     Danbury
4  nyu_2451_34755.5       4  New Canaan

                                            geometry    Operating
0  MULTILINESTRING ((-73.93795 41.70584, -73.9472…  Metro North
1  MULTILINESTRING ((-72.92175 41.30498, -72.9282…  Metro North
2  MULTILINESTRING ((-73.56220 41.81472, -73.5582…  Metro North
3  MULTILINESTRING ((-73.45016 41.39636, -73.4181…  Metro North
4  MULTILINESTRING ((-73.49563 41.14630, -73.4981…  Metro North
```

[27]:
```
mnrail=mnrail.to_crs('epsg:3424')
```

[28]:
```
#combine all three rail line datasets
linjrail = lirail.append(njrail)
nymarail=linjrail.append(mnrail)
```

[29]:
```
#trim columns
columns_to_keep=['linename','geometry','Operating']
nymarail=nymarail[columns_to_keep]
nymarail.sample(10)
```

[29]:
```
                    linename  \
26                      PATH
5                  Waterbury
9             MORRIS & ESSEX
2                  MAIN LINE
29                     SEPTA
15           PATCO SPEEDLINE
1             Port Jefferson
11  NORTH JERSEY COAST LINE
```

```
7            MORRIS & ESSEX
27                     PATH

                                                 geometry             Operating
26  MULTILINESTRING ((621237.511 692081.826, 62123…   New Jersey Railroad
5   MULTILINESTRING ((890037.103 993921.210, 88860…           Metro North
9   LINESTRING (399357.761 735253.132, 399513.899 …   New Jersey Railroad
2   LINESTRING (587611.064 830753.567, 588462.132 …   New Jersey Railroad
29  LINESTRING (301113.611 409819.346, 301275.538 …   New Jersey Railroad
15  LINESTRING (351629.897 364774.694, 351391.896 …   New Jersey Railroad
1   MULTILINESTRING ((657530.427 697249.579, 65739…  Long Island Railroad
11  LINESTRING (618407.609 452679.772, 618718.882 …   New Jersey Railroad
7   LINESTRING (399357.761 735253.132, 399513.899 …   New Jersey Railroad
27  LINESTRING (613236.333 691854.525, 613327.582 …   New Jersey Railroad
```

4.2 Data exploratin for transit stations

[30]:
```
njstation.head()
```

[30]:
```
   OBJECTID    COUNTY   LATITUDE  LONGITUDE         STATION  \
0         1     OCEAN  40.092718 -74.048192  Point Pleasant
1         2  MONMOUTH  40.150567 -74.035460     Spring Lake
2         3  MONMOUTH  40.180589 -74.027296          Belmar
3         4  MONMOUTH  40.203775 -74.018956   Bradley Beach
4         5  MONMOUTH  40.215360 -74.014788     Asbury Park

                 RAIL_LINE                     MUN_LABEL   ATIS_ID AMTRAK  \
0  North Jersey Coast Line  Point Pleasant Beach Borough  RAIL0122      N
1  North Jersey Coast Line           Spring Lake Borough  RAIL0141      N
2  North Jersey Coast Line                Belmar Borough  RAIL0015      N
3  North Jersey Coast Line         Bradley Beach Borough  RAIL0022      N
4  North Jersey Coast Line              Asbury Park City  RAIL0008      N

                        geometry
0  POINT (618521.134 459008.903)
1  POINT (621972.996 480099.144)
2  POINT (624196.751 491047.221)
3  POINT (626480.961 499505.650)
4  POINT (627622.290 503731.988)
```

[31]:
```
columns_to_keep=['STATION','LATITUDE','LONGITUDE','geometry']
njstation=njstation[columns_to_keep]
njstation.columns=['stationname','lat','lon','geometry']
njstation['Operating'] = 'New Jersey Railroad'
njstation.head()
```

[31]:
```
      stationname        lat        lon                       geometry  \
0  Point Pleasant  40.092718 -74.048192  POINT (618521.134 459008.903)
1     Spring Lake  40.150567 -74.035460  POINT (621972.996 480099.144)
2          Belmar  40.180589 -74.027296  POINT (624196.751 491047.221)
3   Bradley Beach  40.203775 -74.018956  POINT (626480.961 499505.650)
4     Asbury Park  40.215360 -74.014788  POINT (627622.290 503731.988)

             Operating
0  New Jersey Railroad
1  New Jersey Railroad
2  New Jersey Railroad
3  New Jersey Railroad
4  New Jersey Railroad
```

[32]:
```
columns_to_keep=['stop_name','stop_lat','stop_lon','geometry']
listation=listation[columns_to_keep]
listation.columns=['stationname','lat','lon','geometry']
listation['Operating'] = 'Long Island Railroad'
listation.head()
```

[32]:
```
           stationname       lat       lon                       geometry  \
0    Long Island City  40.74128 -73.95639  POINT (-73.95639 40.74128)
1  Hunterspoint Avenue  40.74238 -73.94679  POINT (-73.94679 40.74238)
2        Penn Station  40.75058 -73.99358  POINT (-73.99358 40.75058)
3            Woodside  40.74584 -73.90297  POINT (-73.90297 40.74584)
4        Forest Hills  40.71957 -73.84481  POINT (-73.84481 40.71957)

             Operating
0  Long Island Railroad
1  Long Island Railroad
2  Long Island Railroad
3  Long Island Railroad
4  Long Island Railroad
```

[33]:
```
listation=listation.to_crs('epsg:3424')
```

[34]:
```
columns_to_keep=['stop_name','stop_lat','stop_lon','geometry']
mnstation=mnstation[columns_to_keep]
mnstation.columns=['stationname','lat','lon','geometry']
mnstation['Operating'] = 'Metro North Railroad'
mnstation.head()
```

[34]:
```
          stationname        lat        lon                       geometry  \
0       Grand Central  40.752998 -73.977056  POINT (-73.97706 40.75300)
1    Harlem-125th St.  40.805157 -73.939149  POINT (-73.93915 40.80516)
2    Yankees-E153 St.  40.825300 -73.929900  POINT (-73.92990 40.82530)
3      Morris Heights  40.854252 -73.919583  POINT (-73.91958 40.85425)
```

```
4  University Heights  40.862248 -73.913120  POINT (-73.91312 40.86225)

              Operating
0  Metro North Railroad
1  Metro North Railroad
2  Metro North Railroad
3  Metro North Railroad
4  Metro North Railroad
```

[35]:
```
mnstation=mnstation.to_crs('epsg:3424')
```

[36]:
```
#append to combine three station datasets together
linjstation= listation.append(njstation)
nymastation=linjstation.append(mnstation)
nymastation.sample(10)
```

[36]:
```
                         stationname        lat        lon  \
15                         Irvington  41.039993 -73.873083
76                           Boonton  40.903384 -74.407730
5                        Kew Gardens  40.709640 -73.830890
220                    Hamilton Ave.  40.211941 -74.755765
270  Fern Rock Transportation Center  40.041024 -75.134802
37                            Roslyn  40.790470 -73.643260
159                  Woodcliff Lake  41.021078 -74.040772
71                         Greenport  41.099700 -72.363100
16                        Douglaston  40.768060 -73.749410
235              Cooper St./Rutgers  39.947485 -75.124145

                            geometry             Operating
15    POINT (665053.652 804387.677)  Metro North Railroad
76    POINT (517629.100 754011.669)   New Jersey Railroad
5     POINT (677612.026 684123.665)  Long Island Railroad
220   POINT (420698.190 502218.884)   New Jersey Railroad
270   POINT (314400.586 440491.624)   New Jersey Railroad
37    POINT (729338.874 714022.051)  Long Island Railroad
159   POINT (618834.399 797208.711)   New Jersey Railroad
71   POINT (1081051.209 832740.017)  Long Island Railroad
16    POINT (700017.690 705588.664)  Long Island Railroad
235   POINT (317145.778 406398.641)   New Jersey Railroad
```

[37]:
```
nymastation=nymastation.to_crs('epsg:4326')
```

[38]:
```
#trim the county border dataset to NYMA 31 counties
cb = cb[cb.FIPS.isin(["9001",
"9005",
"9009",
"34003",
```

```
"34013",
"34017",
"34019",
"34021",
"34023",
"34025",
"34027",
"34029",
"34031",
"34035",
"34037",
"34039",
"36005",
"36027",
"36047",
"36059",
"36061",
"36071",
"36079",
"36081",
"36085",
"36087",
"36103",
"36111",
"36119",
"42089",
"42103",
])]
```

4.3 Calculate Transit Density

[39]:
```
# sjoin to combine station dataset with county border dataframe
stationincounty = gpd.sjoin(nymastation, cb)
stationincounty.head()
```

[39]:

	stationname	lat	lon	geometry	\
0	Long Island City	40.74128	-73.95639	POINT (-73.95639 40.74128)	
1	Hunterspoint Avenue	40.74238	-73.94679	POINT (-73.94679 40.74238)	
3	Woodside	40.74584	-73.90297	POINT (-73.90297 40.74584)	
4	Forest Hills	40.71957	-73.84481	POINT (-73.84481 40.71957)	
5	Kew Gardens	40.70964	-73.83089	POINT (-73.83089 40.70964)	

	Operating	index_right	id	admin_name	\
0	Long Island Railroad	3138	wg010mf7692.3090	Queens County	
1	Long Island Railroad	3138	wg010mf7692.3090	Queens County	
3	Long Island Railroad	3138	wg010mf7692.3090	Queens County	
4	Long Island Railroad	3138	wg010mf7692.3090	Queens County	

```
5  Long Island Railroad         3138  wg010mf7692.3090  Queens County

  admin_fips state state_fips    name  suffix      pop    sq_miles  \
0      36081    NY         36  Queens  County  2230722  109.789692
1      36081    NY         36  Queens  County  2230722  109.789692
3      36081    NY         36  Queens  County  2230722  109.789692
4      36081    NY         36  Queens  County  2230722  109.789692
5      36081    NY         36  Queens  County  2230722  109.789692

   prim_miles  countyp010   FIPS
0  102.299973        1013  36081
1  102.299973        1013  36081
3  102.299973        1013  36081
4  102.299973        1013  36081
5  102.299973        1013  36081
```

[40]:
```
#count station number in each county and create a new dataframe
stationnumber = stationincounty.FIPS.value_counts().rename_axis('FIPS').
 ↪reset_index(name='station_count')
stationnumber.columns=['FIPS','station_count']
county=cb.merge(stationnumber,on='FIPS')
#calculate transit denisty
county['TransitDensity'] = county['station_count']/county['sq_miles']
county.head()
```

[40]:
```
                 id        admin_name admin_fips state state_fips       name  \
0   wg010mf7692.274  New Haven County      09009    CT         09  New Haven
1  wg010mf7692.1647  Middlesex County      34023    NJ         34  Middlesex
2  wg010mf7692.1648   Monmouth County      34025    NJ         34   Monmouth
3  wg010mf7692.1649     Morris County      34027    NJ         34     Morris
4  wg010mf7692.1650    Passaic County      34031    NJ         34    Passaic

   suffix     pop    sq_miles  prim_miles  countyp010  \
0  County  862477  619.868798  170.575772         824
1  County   10959  314.088885   99.996252        1060
2  County  630380  475.639443  173.523904        1088
3  County   12934  481.387018  131.986283         944
4  County  501226  197.619343   90.313878         911

                                            geometry   FIPS  station_count  \
0  MULTIPOLYGON (((-72.93470 41.61544, -72.93431 …   9009             10
1  MULTIPOLYGON (((-74.29509 40.59449, -74.29389 …  34023             10
2  MULTIPOLYGON (((-73.99727 40.47624, -73.99734 …  34025             14
3  MULTIPOLYGON (((-74.49702 41.03445, -74.49701 …  34027             19
4  MULTIPOLYGON (((-74.23702 41.08693, -74.23809 …  34031              9

   TransitDensity
```

```
0        0.016132
1        0.031838
2        0.029434
3        0.039469
4        0.045542
```

[41]:
```
#drop 5 counties in NYC since they are mostly likely to be outliers
county=county.drop([20, 21,18,19])
```

[42]:
```
#create two new data frame with the top 5 highest transit density and the␣
 ↪bottom 5 lowest transit density
tdt5=county.sort_values(by='TransitDensity',ascending = False).head(5)
tdb5=county.sort_values(by='TransitDensity',ascending = False).tail(5)
#create new dataframe by combining the top 5 and bottom 5 counties for plotting
td=tdt5.append(tdb5)
```

[43]:
```
#put top and bottom 5 counties with highest and lowest transit density on the␣
 ↪graph
figtd=px.bar(
      td,
      x='TransitDensity',
      y='admin_name',
      orientation='h', #change orientation of bar chart
      color='TransitDensity',
      color_continuous_scale='Bluered',
    width=800, height=400#change color of the bars
)
#update bar charts
figtd.update_layout(title={
        'text': "Transit Density", #add title
        'y':1,#change position of the title
        'x':0.5}
)
figtd
```

[44]:
```
figtd1=px.bar(
      td,
      x='TransitDensity',
      y='admin_name',
      orientation='h', #change orientation of bar chart
      color='TransitDensity',
      color_continuous_scale='Bluered',#change color of the bars
)
#update bar charts
figtd1.update_layout(title={
        'text': "Transit Density", #add title
        'y':1,#change position of the title
```

```
        'x':0.5}
)
figtd1
```

[45]:
```
figtd.write_html("toptransitdensity.html")
```

[46]:
```
figtd1.write_html("toptransitdensitybig.html")
```

4.4 Transit Density and Migration Correlation Analysis

Now we have the transit density data for each county, we want to know how does transit density relate to migration? We explore the correlation between the two factors by using the correlation analysis in Plotly Express.

We created a csv file with the data from the analysis on migration, transit density, housing, economic and demographic charateristics for the simplicity purpose to analyze the correlations between migration and transit density. We also used the same dataframe in other notebooks for a more holistic analysis of correlations.

[47]:
```
#plot linear relationship between transit station and transit density with␣
↪migration

#import metrics data frame we created from analyzing migration, transit␣
↪density, housing, demogrphic and economic characteristics
metric = pd.read_csv('metric.csv')
metric.head()
```

[47]:

	FIPS	Geographic Area Name	Station Count	Outflow_2018	\
0	9001	Fairfield County, Connecticut	0	15281	
1	9005	Litchfield County, Connecticut	0	2752	
2	9009	New Haven County, Connecticut	10	8883	
3	34003	Bergen County, New Jersey	31	20899	
4	34013	Essex County, New Jersey	39	20566	

	Inflow_2018	TransitDensity	Total Population	Commuter	HomeWorker	\
0	19508	0.0432	0.0108	0.2074	0.1525	
1	3866	0.0000	-0.0241	-0.0727	0.0545	
2	11607	0.0161	-0.0044	0.0422	0.3057	
3	21226	0.1284	0.0104	0.3594	0.0523	
4	19458	0.3058	0.0050	0.3556	0.3090	

	Median Age	…	GDP	Jobs	Income Level	GDPvsIncome	OwnedUnits	\
0	0.0177	…	0.0966	0.0263	0.1655	0.6904	0.0037	
1	0.0396	…	0.0522	-0.0055	-0.0095	0.7605	-0.0180	
2	0.0126	…	0.1342	0.0290	0.2862	1.7877	-0.0109	
3	0.0097	…	0.1274	0.0427	0.3661	0.8718	0.0021	
4	0.0218	…	0.1395	0.0571	0.4526	1.4619	-0.0045	

```
   RentalUnits  Values   Rents  OwnedAffordability  RentalAffordability
0       0.0008 -0.0002  0.1078              0.1092              -0.0132
1      -0.0852 -0.0319  0.0951              0.1319              -0.0019
2      -0.0315 -0.0148  0.0648              0.1032               0.1017
3      -0.0410  0.0354  0.0851              0.1383               0.0663
4      -0.0103  0.0405  0.0692              0.1577               0.0091

[5 rows x 22 columns]
```

[48]:
```
#Correlation analysis using OLS
fig2 = px.
 ↪scatter(metric,x='TransitDensity',y=['Inflow_2018','Outflow_2018'],trendline="ols",labels={
                      "Inflow_2018": "In-Migration Population","value":
 ↪"Migration Population"})
fig2.update_layout(title={
        'text': "Correlation between Station Count and Migration", #add title
        'y':0.95,#change position of the title
        'x':0.5}
)
fig2
```

[49]:
```
results = px.get_trendline_results(fig2)
results = results.iloc[0]["px_fit_results"].summary()
print(results)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.234
Model:                            OLS   Adj. R-squared:                  0.202
Method:                 Least Squares   F-statistic:                     7.334
Date:                Mon, 15 Mar 2021   Prob (F-statistic):             0.0123
Time:                        04:21:34   Log-Likelihood:                -265.92
No. Observations:                  26   AIC:                             535.8
Df Residuals:                      24   BIC:                             538.4
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       1.094e+04   1551.446      7.052      0.000    7738.408    1.41e+04
x1          2.629e+04   9707.256      2.708      0.012    6253.329    4.63e+04
==============================================================================
Omnibus:                        2.379   Durbin-Watson:                   1.089
Prob(Omnibus):                  0.304   Jarque-Bera (JB):                1.646
Skew:                           0.407   Prob(JB):                        0.439
Kurtosis:                       2.075   Cond. No.                         7.15
```

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

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

The correlation plot showed that there is a positive relationship between transit density and migration in both directions, meaning transit density can attract populaiton migration into a county and also drive outmigration. The outmigration has a stronger correlation with transit density, which indicates that one unit increase in transit density can drive more people out of a county than attract people to move into a county. Both outflow and inflow are significantly correlated with transit density.

[50]:
```
#save the plot as html
fig2.write_html("transitdensitymigration.html")
```

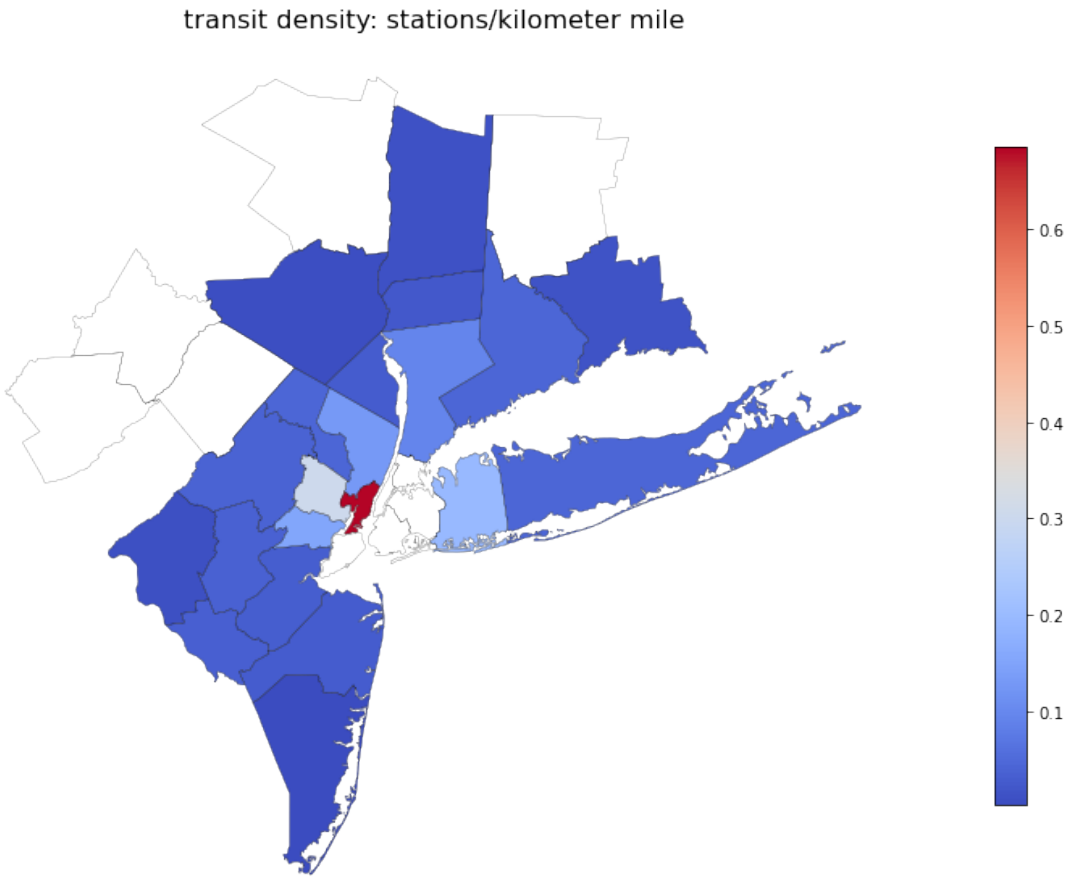
4.5 Visualization of transit density and transit stations in New York Metro Area

In this section, we experimented with various ways to visualize the transit density data, we want to create a map that the audiences can intercat with and get the information they want. We created the basic choropleth map, an interactive folium map, a plotly scatterplot to show various options to visualize the data.

We also conducted point analysis to further our understanding of the transit station pattern in New York Metro Area.

[51]:
```
#choropleth map for transit density
fig, ax = plt.subplots(figsize = (20,10))
county.plot(column = 'TransitDensity',
        cmap = 'coolwarm',
        legend=True,
        ax=ax,
        legend_kwds={'shrink': 0.75}
         )
cb.geometry.plot(facecolor="none",edgecolor='black',linewidth = 0.2,ax=ax)
plt.axis("off")
plt.title("transit density: stations/kilometer mile", fontsize=16)
```

[51]:
```
Text(0.5, 1.0, 'transit density: stations/kilometer mile')
```



It's very simple to show the transit density on a choropleth map, but audiences can't see how many stations are in each county and their locations. So we decided to create an interactive map that shows the location of each transit stations for a better representation of the New York Metro Area's transit environment.

4.5.1 Folium Map

[52]:
```
#get the mean latitude for folium map
latitude = nymastation.lat.mean()
latitude
```

[52]: 40.7328522284504

[53]:
```
#get the mean longitude for folium map
longitude = nymastation.lon.mean()
longitude
```

[53]: -74.02803319332598

[54]:
```
#create the base folium map
m = folium.Map(location=[latitude,longitude], zoom_start=13)
m
```

[54]:
```
<folium.folium.Map at 0x7ff4d05c8490>
```

[55]:
```
#create folium map to show the geogrphaic concentration of transit stations
m = folium.Map(location=[latitude,longitude], tiles='CartoDB dark_matter',␣
 ↪zoom_start=13)

marker_cluster = MarkerCluster().add_to(m)

for index, row in nymastation.iterrows():
    tooltip_text = row.stationname + ', operated by ' + row.Operating
    folium.Marker([row.lat,row.lon],
                  popup=row.stationname,
                  tooltip=tooltip_text,).add_to(marker_cluster)
m
```

[55]:
```
<folium.folium.Map at 0x7ff4d0698e80>
```

As expected, there is a concentration of transit station in New York City, and the concentration decreases as the distance between an area with NYC increases.

[56]:
```
#save the folium map
m.save('transit stations.html')
```

[57]:
```
factor = pd.read_csv('Combined_Census.csv')
factor['FIPS'] = factor['AFFGEOID'].str.strip().str[-5:]
factor.head()
```

[57]:

	AFFGEOID	Geographic Area Name	Out-Migration	\
0	0500000US34017	Hudson County, New Jersey	2849	
1	0500000US34029	Ocean County, New Jersey	-371	
2	0500000US09001	Fairfield County, Connecticut	801	
3	0500000US34003	Bergen County, New Jersey	1359	
4	0500000US34039	Union County, New Jersey	1201	

	Transit Density	Railway Transportation	Total Population	Work From Home	\
0	0.6851	0.0330	13753	0.0602	
1	0.0031	-0.0491	10526	0.0507	
2	0.0432	0.3556	10133	0.3090	
3	0.1284	0.1696	9543	0.1746	
4	0.1541	-0.0957	7830	0.0495	

	GDP	Number of Jobs	Income Level	\
0	0.1475	0.0441	0.6609	

```
1  0.1504          0.0385        0.0795
2  0.1395          0.0571        0.4526
3  0.1450          0.0584        0.6739
4  0.2400          0.0723        1.0136

   Housing affordability % change for owner-occupied units  \
0                                                  8.0
1                                                  7.0
2                                                  6.0
3                                                  8.0
4                                                  8.0

   Housing affordability % change for rental units  \
0                                              0.0
1                                              5.0
2                                             -1.0
3                                              3.0
4                                              4.0

   Median Number of Owned-Units  Median Rent   FIPS
0                       -0.0077       0.1178  34017
1                       -0.0089       0.0710  34029
2                       -0.0045       0.0692  09001
3                        0.0135       0.1148  34003
4                        0.0674       0.1094  34039
```

[58]:
```
factor.dtypes
```

[58]:
```
AFFGEOID                                                 object
Geographic Area Name                                     object
Out-Migration                                             int64
Transit Density                                         float64
Railway Transportation                                  float64
Total Population                                          int64
Work From Home                                          float64
GDP                                                     float64
Number of Jobs                                          float64
Income Level                                            float64
Housing affordability % change for owner-occupied units  float64
Housing affordability % change for rental units          float64
Median Number of Owned-Units                            float64
Median Rent                                             float64
FIPS                                                     object
dtype: object
```

[59]:
```
factor['FIPS'] = factor['FIPS'].astype(str).astype(int)
```

[60]:
```
demo = cb.merge(factor,on='FIPS',how='left')
demo.head()
```

[60]:

	id	admin_name	admin_fips	state	state_fips	\
0	wg010mf7692.273	Litchfield County	09005	CT	09	
1	wg010mf7692.274	New Haven County	09009	CT	09	
2	wg010mf7692.1647	Middlesex County	34023	NJ	34	
3	wg010mf7692.1648	Monmouth County	34025	NJ	34	
4	wg010mf7692.1649	Morris County	34027	NJ	34	

	name	suffix	pop	sq_miles	prim_miles	…	\
0	Litchfield	County	189927	944.531099	143.390210	…	
1	New Haven	County	862477	619.868798	170.575772	…	
2	Middlesex	County	10959	314.088885	99.996252	…	
3	Monmouth	County	630380	475.639443	173.523904	…	
4	Morris	County	12934	481.387018	131.986283	…	

	Railway Transportation	Total Population	Work From Home	GDP	\
0	0.3378	-4511.0	0.1854	0.0835	
1	0.0893	-3809.0	0.1997	0.1238	
2	-0.0478	2652.0	0.1152	0.2122	
3	0.1318	-6315.0	-0.0175	0.1514	
4	0.2091	-2720.0	0.1515	-0.0051	

	Number of Jobs	Income Level	\
0	0.0500	0.2633	
1	0.0684	0.3806	
2	0.0645	0.2652	
3	0.0642	0.2821	
4	0.0757	0.2323	

	Housing affordability % change for owner-occupied units	\
0	7.0	
1	6.0	
2	8.0	
3	7.0	
4	7.0	

	Housing affordability % change for rental units	\
0	0.0	
1	5.0	
2	0.0	
3	3.0	
4	3.0	

	Median Number of Owned-Units	Median Rent
0	0.0024	0.0567

```
1                      -0.0138       0.1056
2                      -0.0074       0.0580
3                      -0.0314       0.0863
4                       0.0111       0.1008

[5 rows x 27 columns]
```

4.5.2 Plotly scatter plot

We want to analyze the transit density in each county and create an interactive charts based on the county level, so we adopt the plotly scatter plot to achieve this.

[62]:
```
#get the center point for each polygon
county['cents'] = county.centroid
county.head()
```

```
<ipython-input-62-a468ffa6f2e6>:2: UserWarning:

Geometry is in a geographic CRS. Results from 'centroid' are likely incorrect.
Use 'GeoSeries.to_crs()' to re-project geometries to a projected CRS before this
operation.
```

[62]:

	id	admin_name	admin_fips	state	state_fips	name	\
0	wg010mf7692.274	New Haven County	09009	CT	09	New Haven	
1	wg010mf7692.1647	Middlesex County	34023	NJ	34	Middlesex	
2	wg010mf7692.1648	Monmouth County	34025	NJ	34	Monmouth	
3	wg010mf7692.1649	Morris County	34027	NJ	34	Morris	
4	wg010mf7692.1650	Passaic County	34031	NJ	34	Passaic	

	suffix	pop	sq_miles	prim_miles	countyp010	\
0	County	862477	619.868798	170.575772	824	
1	County	10959	314.088885	99.996252	1060	
2	County	630380	475.639443	173.523904	1088	
3	County	12934	481.387018	131.986283	944	
4	County	501226	197.619343	90.313878	911	

	geometry	FIPS	station_count	\
0	MULTIPOLYGON (((-72.93470 41.61544, -72.93431 …	9009	10	
1	MULTIPOLYGON (((-74.29509 40.59449, -74.29389 …	34023	10	
2	MULTIPOLYGON (((-73.99727 40.47624, -73.99734 …	34025	14	
3	MULTIPOLYGON (((-74.49702 41.03445, -74.49701 …	34027	19	
4	MULTIPOLYGON (((-74.23702 41.08693, -74.23809 …	34031	9	

	TransitDensity	cents
0	0.016132	POINT (-72.93208 41.41072)

```
1        0.031838  POINT (-74.41302 40.43840)
2        0.029434  POINT (-74.22570 40.25891)
3        0.039469  POINT (-74.54442 40.86198)
4        0.045542  POINT (-74.30012 41.03372)
```

[63]:
```
#create two new columns in the county dataframe with x and y coordinates
county['x'] = county.cents.x
county['y'] = county.cents.y
```

[64]:
```
#use plotly to create the map
density = px.scatter_geo(county,
                         lon="x",
                         lat="y",
                         size='TransitDensity',
                         color='TransitDensity',
                         hover_data=['TransitDensity'],
                         size_max=40,
                         scope='usa',
                         hover_name='admin_name',
                         color_continuous_scale = 'sunset')
density.update_geos(fitbounds="locations")
density.show()
```

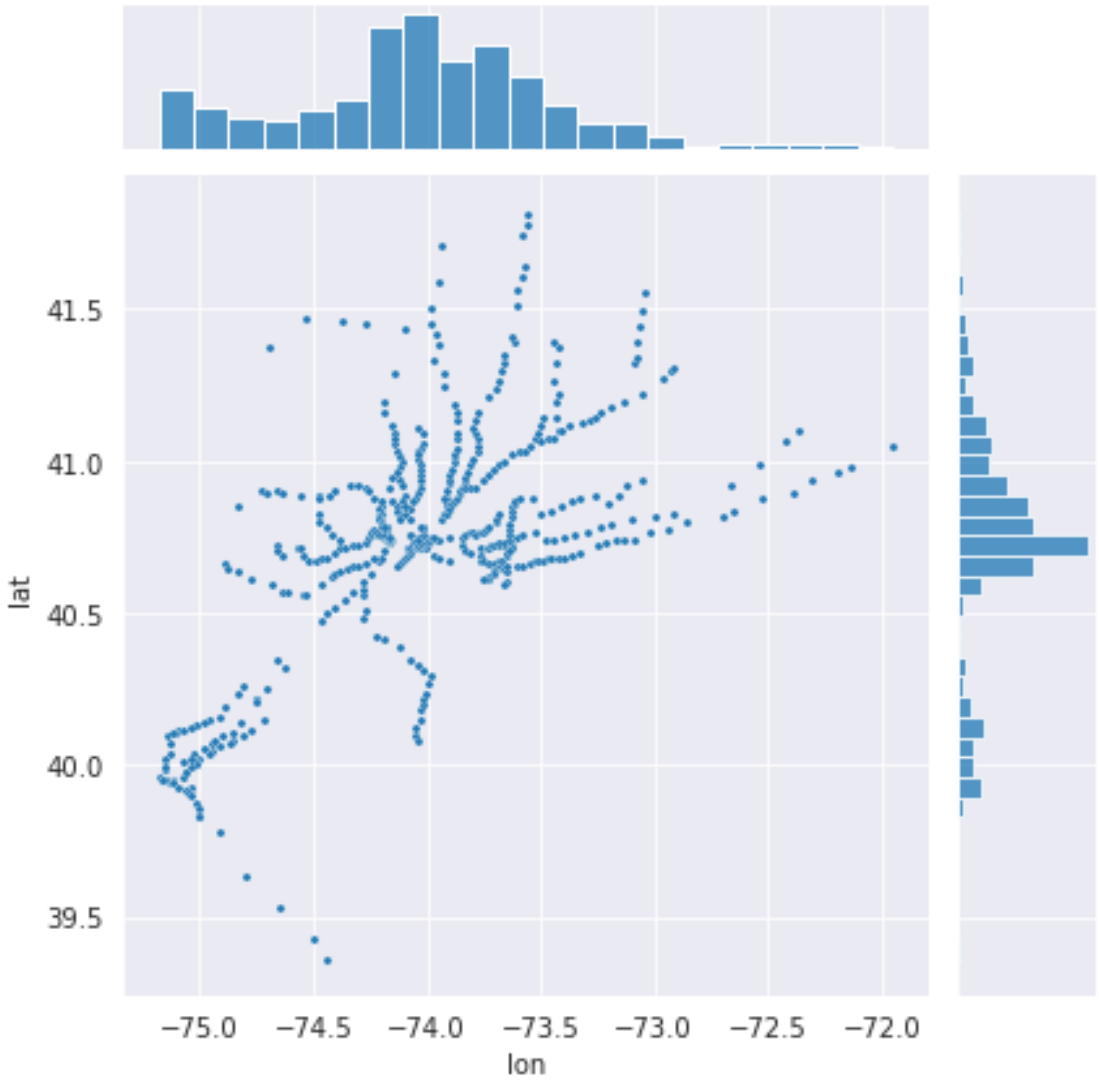
[65]:
```
#save the plotly scatterplot
density.write_html("transitdensity.html")
```

From the plotly scatterplot, we drew the same conclusion as the folium map and the choropleth map. However, in this scatterplot, we excluded 5 counties in New York City since they are outliers for transit density. The plotly scatterplot allows the users to interact with the map.

4.5.3 Transit Station Point Analysis

We are curious about the geographic distribution of transit stations in the New York Metro Area, where we can also see the density of transit station. In this section, we will use the point analysis to document those elements.

[66]:
```
#seaborn map to plot transit station points.
sns.set_style('darkgrid') #change the style of the plot
g = sns.jointplot(data = nymastation,
                  x='lon',
                  y='lat',
                  s=10)
```



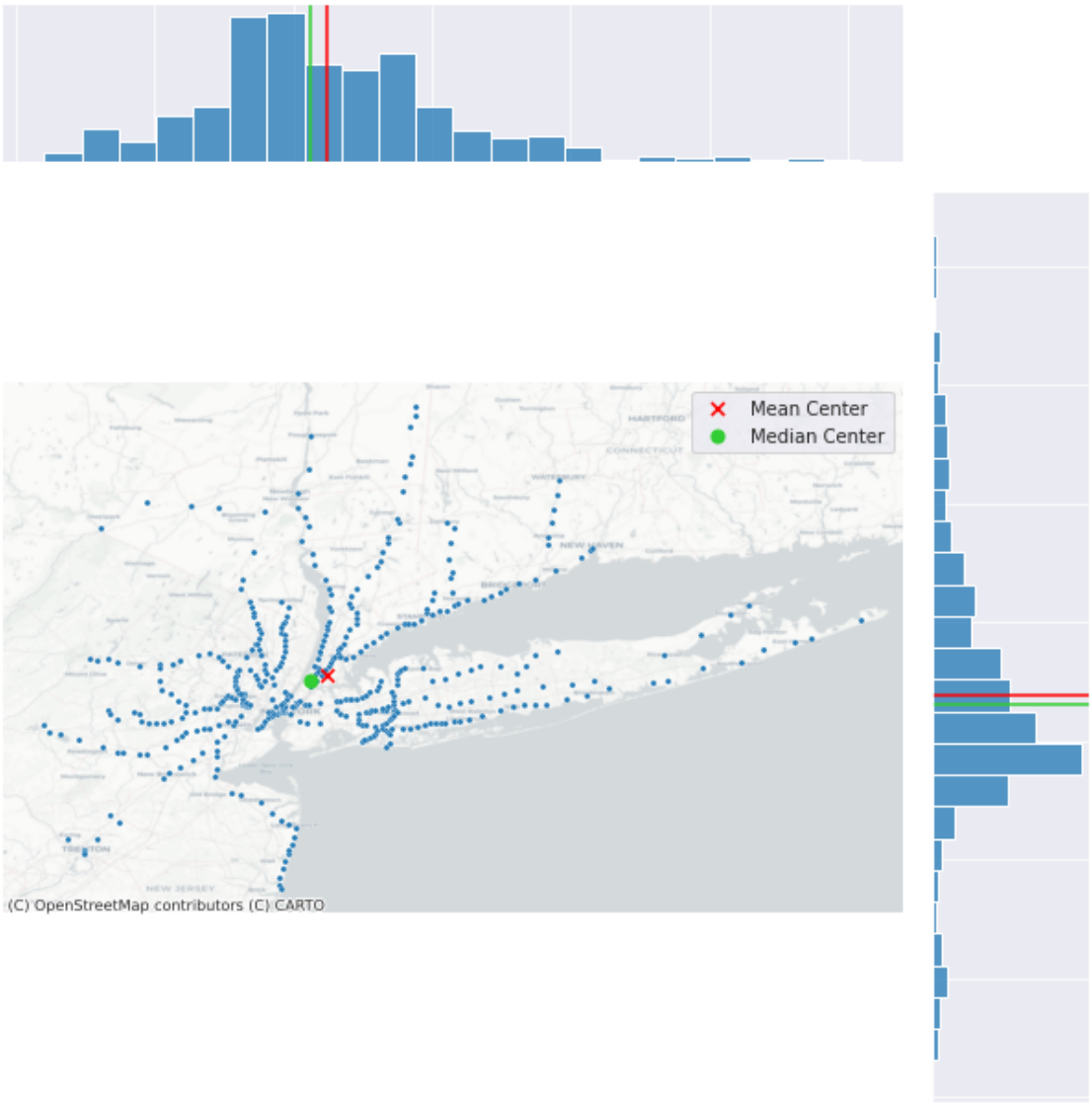
[67]:
```
#KDE plot for transit stations
g = sns.jointplot(data = stationincounty,
                  x='lon',
                  y='lat',
                  kind='kde',
                  shade=True,
                  alpha=0.55,
                  cmap='plasma_r',#change the color
                  n_levels=40,
                  height=10, # make the map bigger
                  )
# Clean axes
g.ax_joint.set_axis_off()
```

```
# add basemap
ctx.add_basemap(g.ax_joint,
                crs='epsg:4326',
                source=ctx.providers.CartoDB.Positron)
```

[68]:
```
#centrography analysis
mean_center0 = centrography.mean_center(stationincounty[['lon','lat']])
med_center0 = centrography.euclidean_median(stationincounty[['lon','lat']])
```

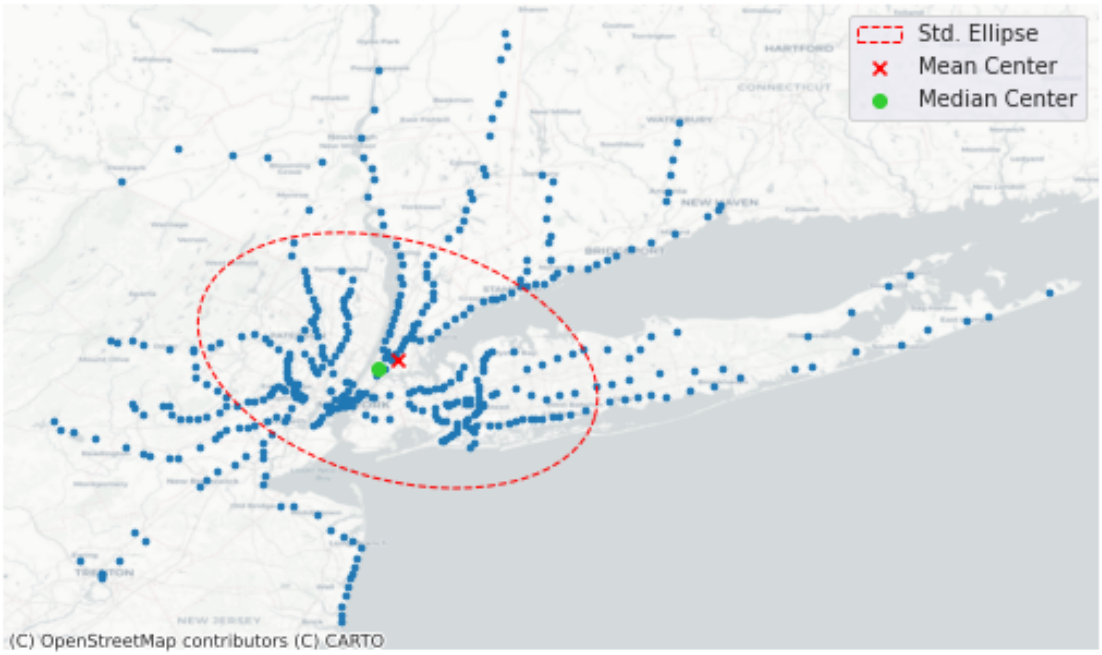
[69]:
```
#put mean and median points on the map
g = sns.jointplot(
```

```
    x='lon', y='lat', data=stationincounty, s=10, height=9
)

# Add mean point and marginal lines
g.ax_joint.scatter(
    *mean_center0, color='red', marker='x', s=50, label='Mean Center'
)
g.ax_marg_x.axvline(mean_center0[0], color='red')
g.ax_marg_y.axhline(mean_center0[1], color='red')

# Add median point and marginal lines
g.ax_joint.scatter(
    *med_center0, color='limegreen', marker='o', s=50, label='Median Center'
)
g.ax_marg_x.axvline(med_center0[0], color='limegreen')
g.ax_marg_y.axhline(med_center0[1], color='limegreen')

# Legend
g.ax_joint.legend()

# Add basemap
ctx.add_basemap(
    g.ax_joint,
    crs='epsg:4326',
    source=ctx.providers.CartoDB.Positron
)

# Clean axes
g.ax_joint.set_axis_off()

# Display
plt.show()
```



[70]:
```
major, minor, rotation = centrography.ellipse(stationincounty[['lon','lat']])
# Set up figure and axis
f, ax = plt.subplots(1, figsize=(9, 9))

ax.scatter(stationincounty['lon'], stationincounty['lat'], s=7)
ax.scatter(*mean_center0, color='red', marker='x', label='Mean Center')
ax.scatter(*med_center0, color='limegreen', marker='o', label='Median Center')

# Construct the standard ellipse using matplotlib
ellipse = Ellipse(xy=mean_center0, # center the ellipse on our mean center
                  width=major*2, # centrography.ellipse only gives half the axis
                  height=minor*2,
```

```
                  angle = numpy.rad2deg(rotation), # Angles for this are in␣
 ↪degrees, not radians
                  facecolor='none',
                  edgecolor='red',
                  linestyle='--',
                  label='Std. Ellipse')
ax.add_patch(ellipse)

ax.legend()

ax.axis('off')

# add a basemap
ctx.add_basemap(ax,
                crs='epsg:4326',
                source=ctx.providers.CartoDB.Positron)

# Display
plt.show()
```



The standard deviation eclipse shows the spatial spread of points on the map. The maximum spread of transit stations is roughly in the northwestern-southeastern direction, and the minimum spread of transit stations is roughly in the southwester-northeastern direction.