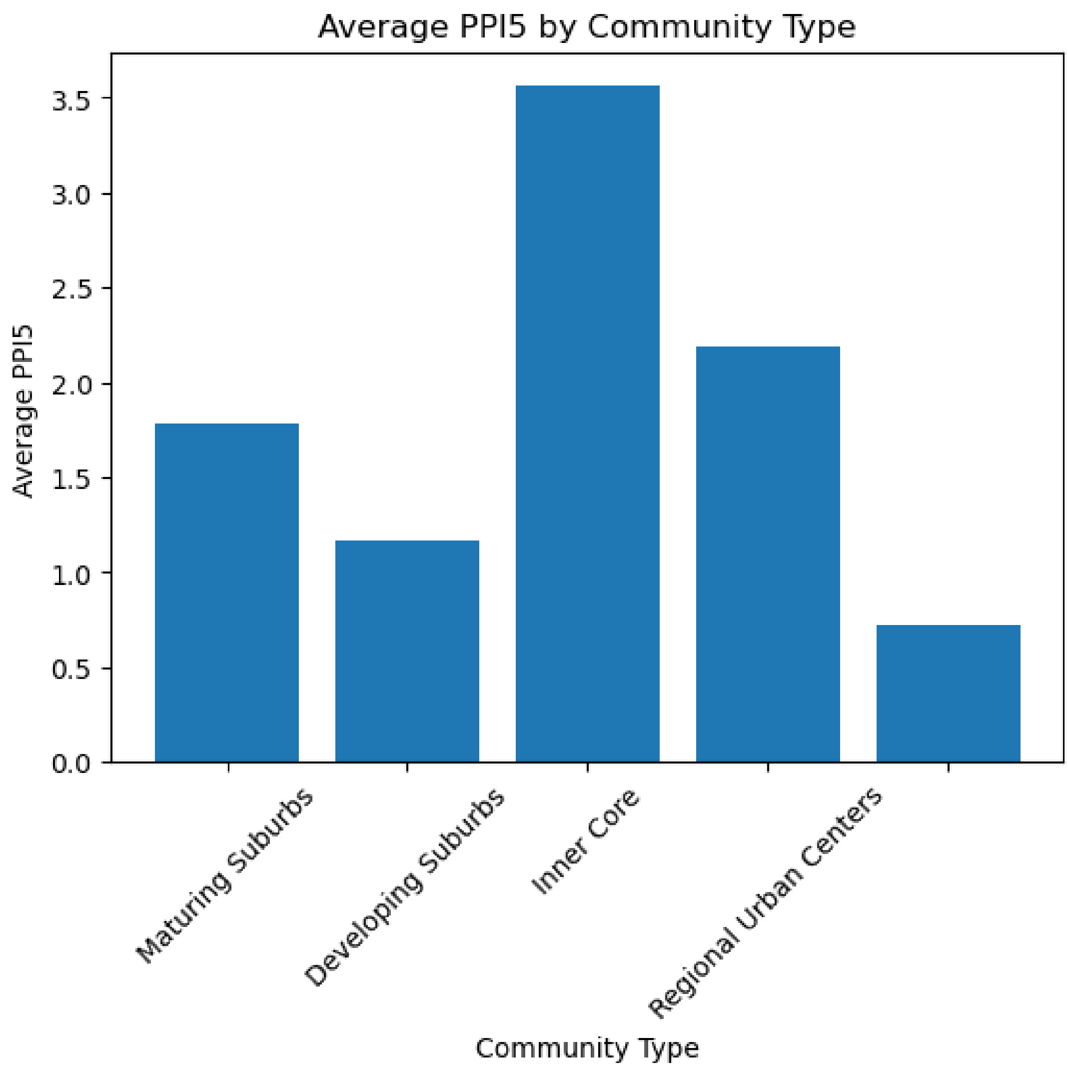
deliverable1submission

November 1, 2023

[19]:
```
# DELIVERABLE 1 TEAM B
# Cathy Wang, Jackson Chiu, Sammy Terada, Naveen Vaidyamath, Jonathan Suarez

#Preliminary Analysis of the batch of data we collected
```

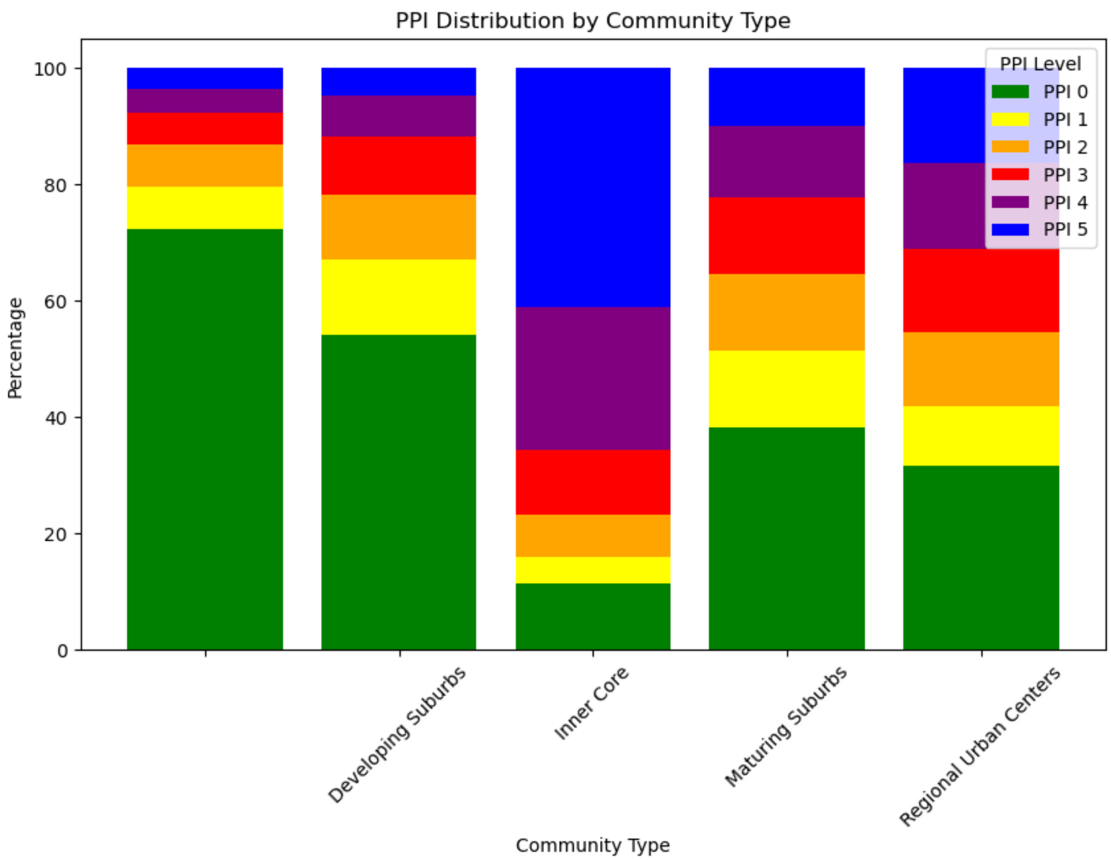
[4]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
 ↪ds-boston-transit-air-quality/fa23-team-b/data/PPI_data.csv')
grouped_data = df.groupby('commtype')['ppi5'].mean()
commtype_order = ['Maturing Suburbs', 'Developing Suburbs', 'Inner Core',␣
 ↪'Regional Urban Centers', ' ']

plt.bar(commtype_order, grouped_data[commtype_order])
plt.xlabel('Community Type')
plt.ylabel('Average PPI5')
plt.title('Average PPI5 by Community Type')
plt.xticks(rotation=45)
plt.show()
```



[3]:

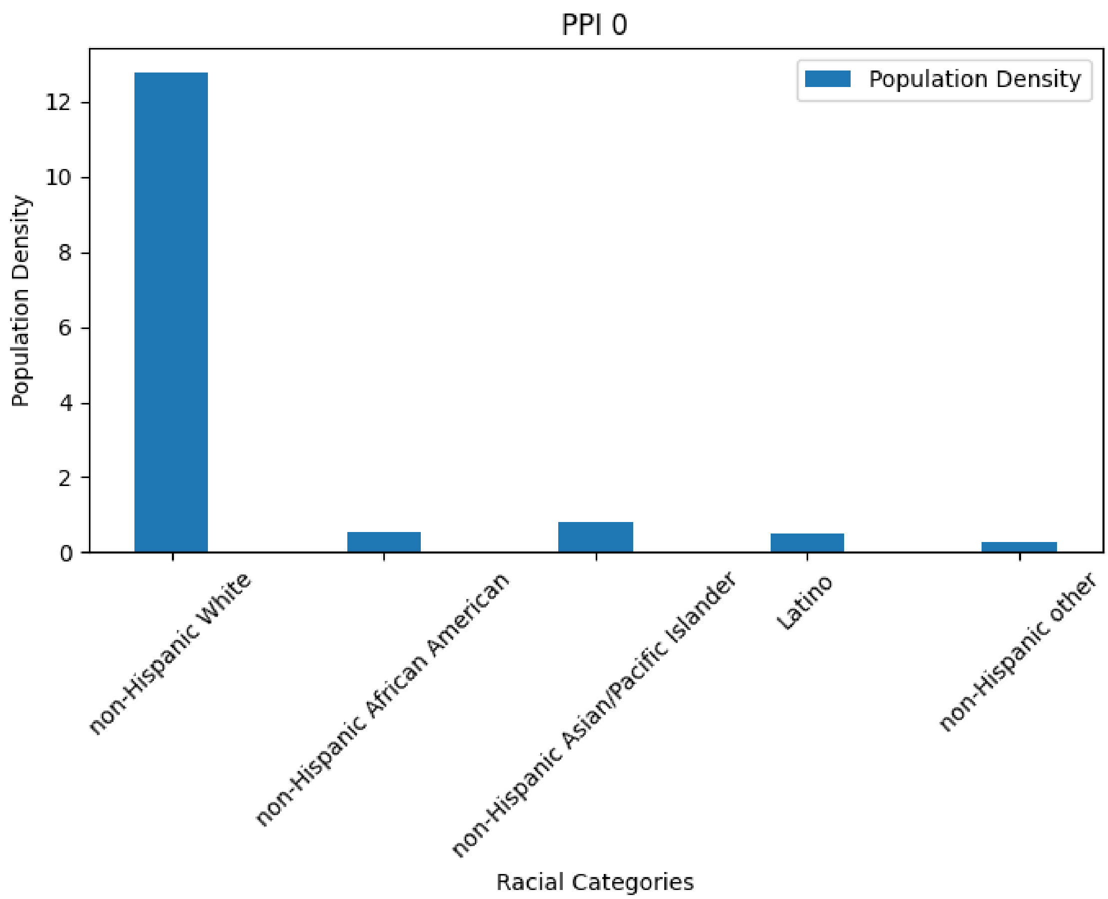
```
colors = ['green', 'yellow', 'orange', 'red', 'purple', 'blue']

stacked_data = df[df['commtype'].isin(commtype_order)][['commtype', 'ppi5']]
stacked_data = stacked_data.pivot_table(index='commtype', columns='ppi5',␣
 ↪aggfunc=len, fill_value=0)

stacked_data = stacked_data.div(stacked_data.sum(axis=1), axis=0) * 100

plt.figure(figsize=(10, 6))
bottom = None
for i, color in enumerate(colors):
    plt.bar(
        stacked_data.index,
        stacked_data[i],
```

```
        label=f'PPI {i}',
        color=color,
        bottom=bottom,
    )
    if bottom is None:
        bottom = stacked_data[i].values
    else:
        bottom += stacked_data[i].values

plt.xlabel('Community Type')
plt.ylabel('Percentage')
plt.title('PPI Distribution by Community Type')
plt.xticks(rotation=45)
plt.legend(title='PPI Level', loc='upper right')

plt.show()
```

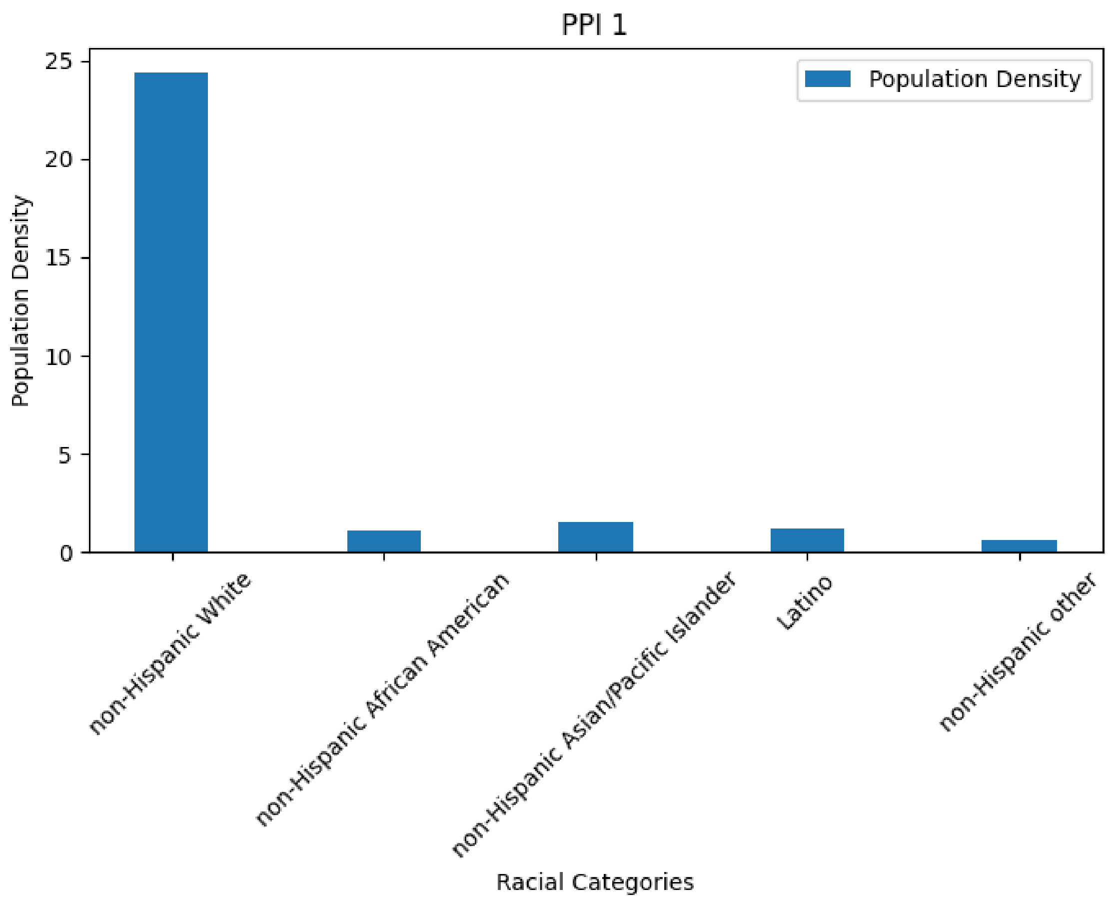



```
[ ]: df = df.rename(columns={'nhwhi_10': 'non-Hispanic White', 'nhaa_10':␣
     ↪'non-Hispanic African American',
```

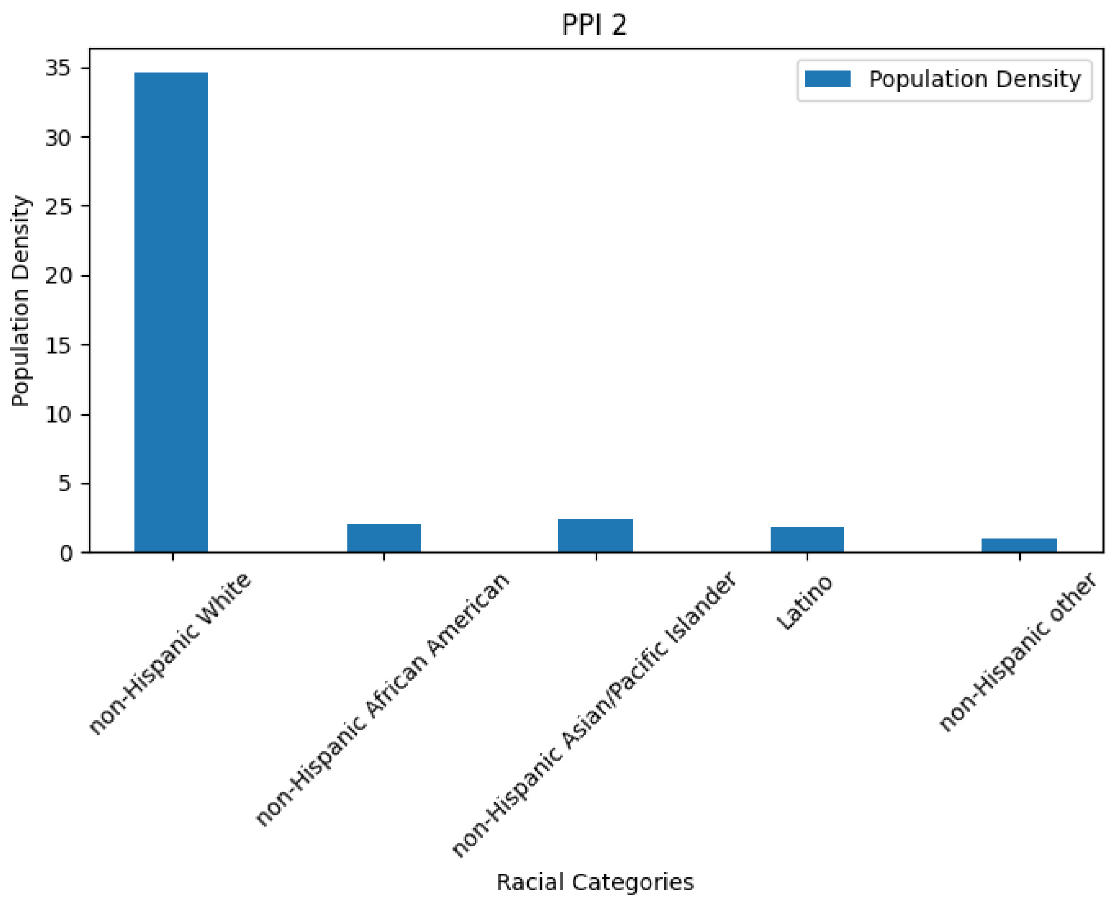
```
                        'nhapi_10': 'non-Hispanic Asian/Pacific␣
↪Islander','lat_10': 'Latino',
                        'nhoth_10': 'non-Hispanic other'})
racial_categories = ['non-Hispanic White', 'non-Hispanic African American',
                                          'non-Hispanic Asian/Pacific␣
↪Islander', 'Latino', 'non-Hispanic other']

fig, ax = plt.subplots(figsize=(8, 4))

zero_avg =df[df['ppi5'] == 0][racial_categories].mean()

bar_width = 0.35
index = range(len(zero_avg))
labels = zero_avg.index
ax.bar(index, zero_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 0')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```



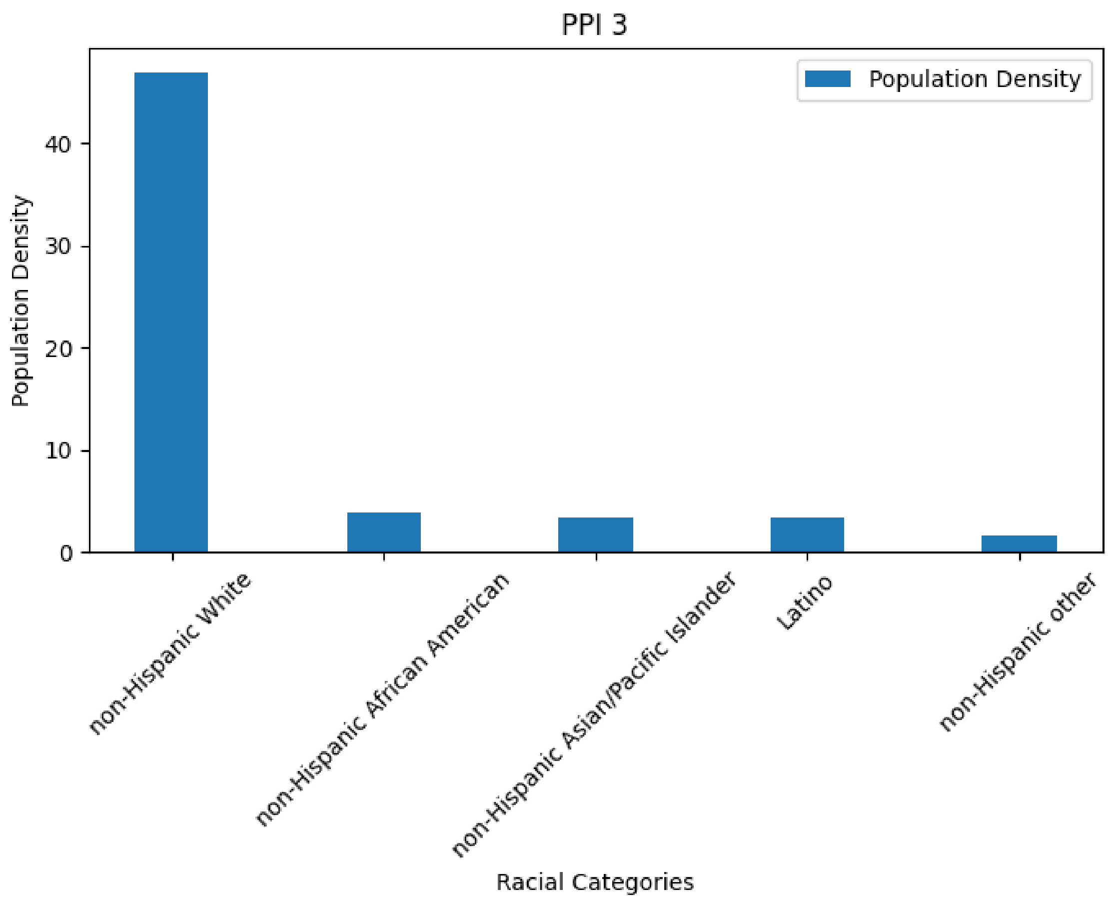
[ ]:
```
fig, ax = plt.subplots(figsize=(8, 4))

one_avg =df[df['ppi5'] == 1][racial_categories].mean()

bar_width = 0.35
index = range(len(one_avg))
labels = one_avg.index
ax.bar(index, one_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 1')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```



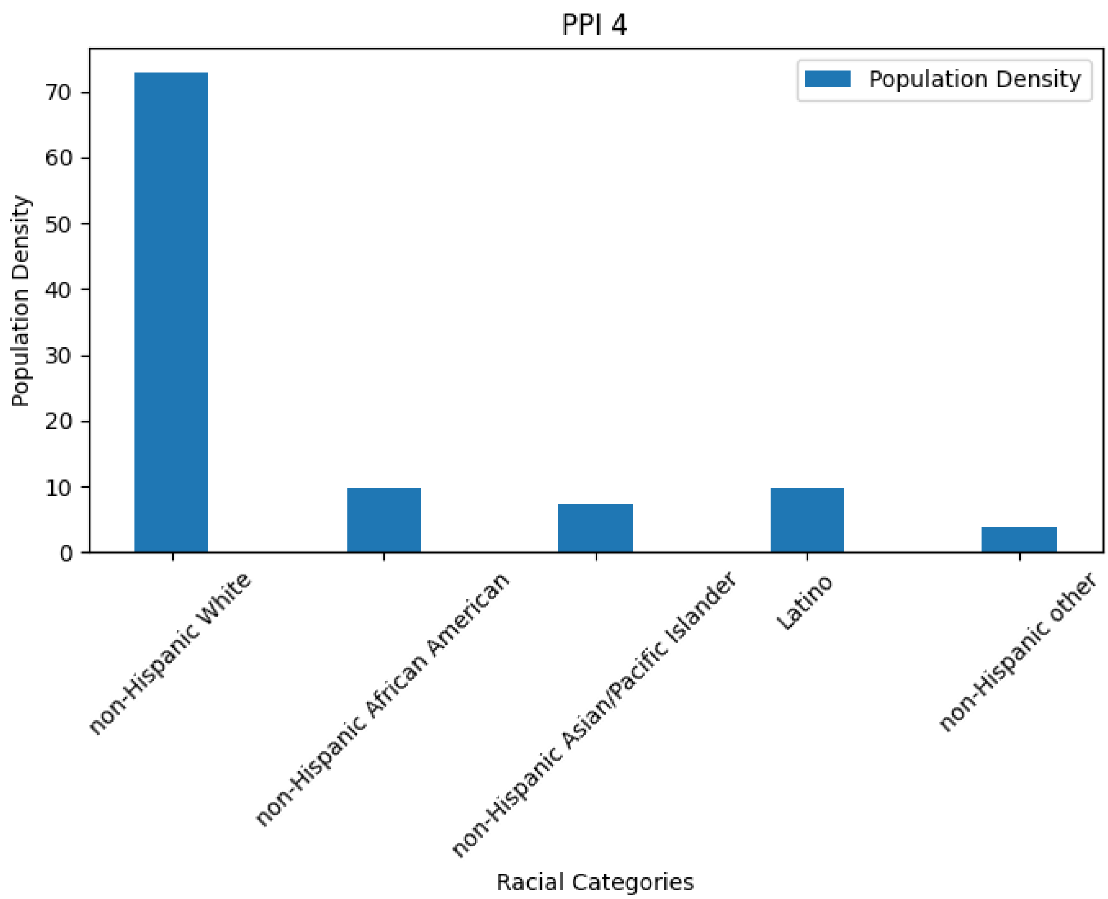
[ ]:
```
fig, ax = plt.subplots(figsize=(8, 4))

two_avg =df[df['ppi5'] == 2][racial_categories].mean()

bar_width = 0.35
index = range(len(two_avg))
labels = two_avg.index
ax.bar(index, two_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 2')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```



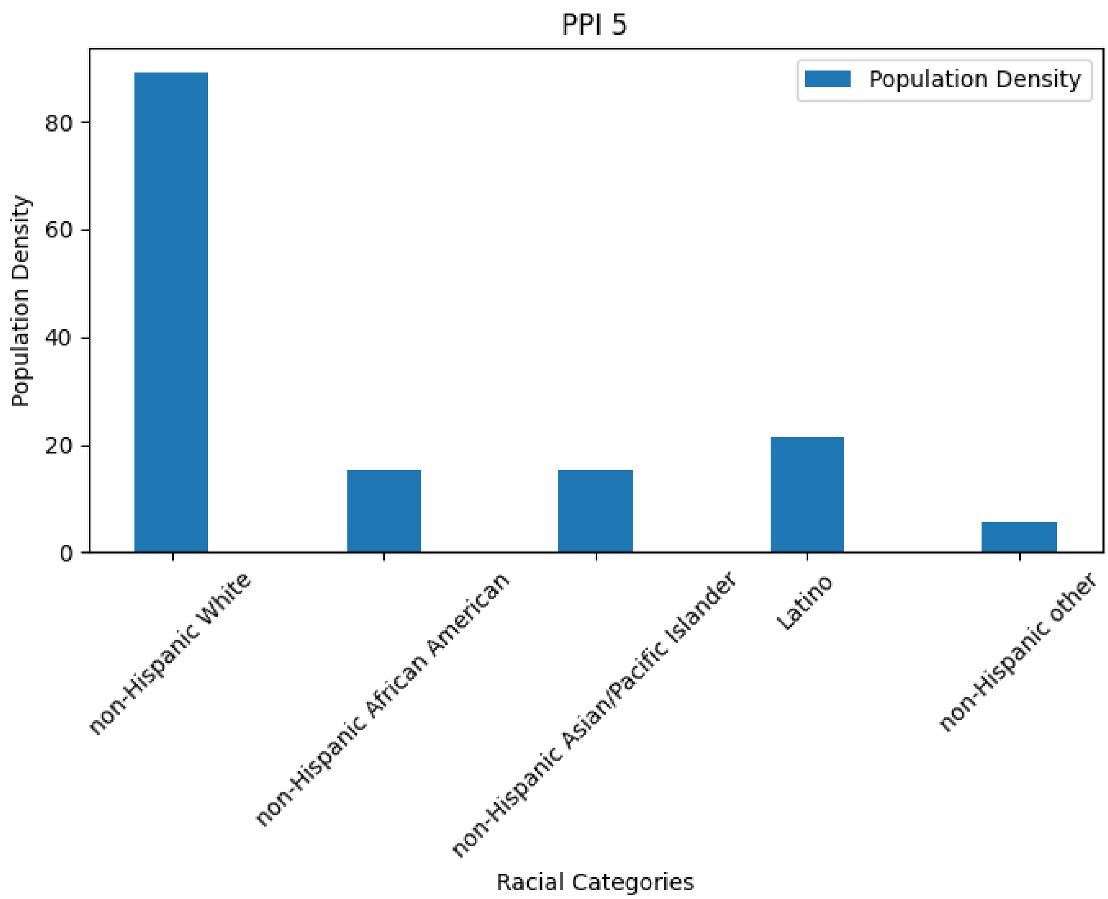
[ ]:
```
fig, ax = plt.subplots(figsize=(8, 4))

three_avg =df[df['ppi5'] == 3][racial_categories].mean()

bar_width = 0.35
index = range(len(three_avg))
labels = three_avg.index
ax.bar(index, three_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 3')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```



[ ]:
```
fig, ax = plt.subplots(figsize=(8, 4))

four_avg =df[df['ppi5'] == 4][racial_categories].mean()

bar_width = 0.35
index = range(len(four_avg))
labels = four_avg.index
ax.bar(index, four_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 4')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```

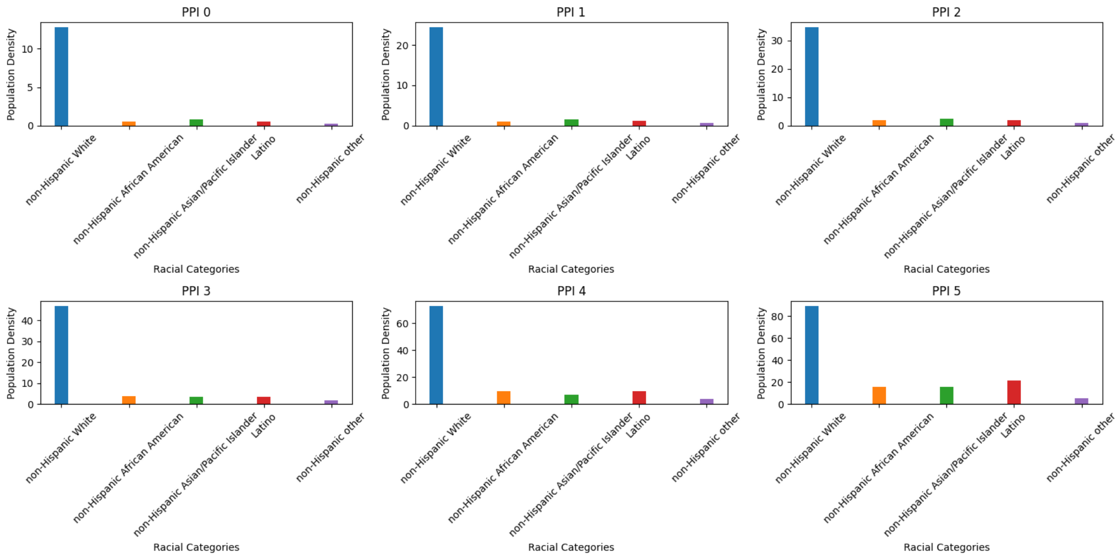


[ ]:
```
fig, ax = plt.subplots(figsize=(8, 4))

five_avg =df[df['ppi5'] == 5][racial_categories].mean()

bar_width = 0.35
index = range(len(five_avg))
labels = five_avg.index
ax.bar(index, five_avg, bar_width, label='Population Density')
ax.set_xlabel('Racial Categories')
ax.set_ylabel('Population Density')
ax.set_title(f'PPI 5')
ax.set_xticks(index)
ax.set_xticklabels(labels, rotation=45)
ax.legend()

plt.show()
```



[ ]:
```
ppi_values = [0, 1, 2, 3, 4, 5]

fig, axes = plt.subplots(2, 3, figsize=(16, 8))

for i, ppi_value in enumerate(ppi_values):
    ppi_df_avg = df[df['ppi5'] == ppi_value][racial_categories].mean()

    row = i // 3
    col = i % 3
    ax = axes[row, col]

    bar_width = 0.2
    index = np.arange(len(ppi_df_avg.index))
    labels = ppi_df_avg.index

    for j, label in enumerate(labels):
        ax.bar(index[j], ppi_df_avg[label], width=bar_width, label=label)

    ax.set_xlabel('Racial Categories')
```

```python
    ax.set_ylabel('Population Density')
    ax.set_title(f'PPI {ppi_value}')
    ax.set_xticks(index)
    ax.set_xticklabels(labels, rotation=45)

plt.tight_layout()
plt.show()
```

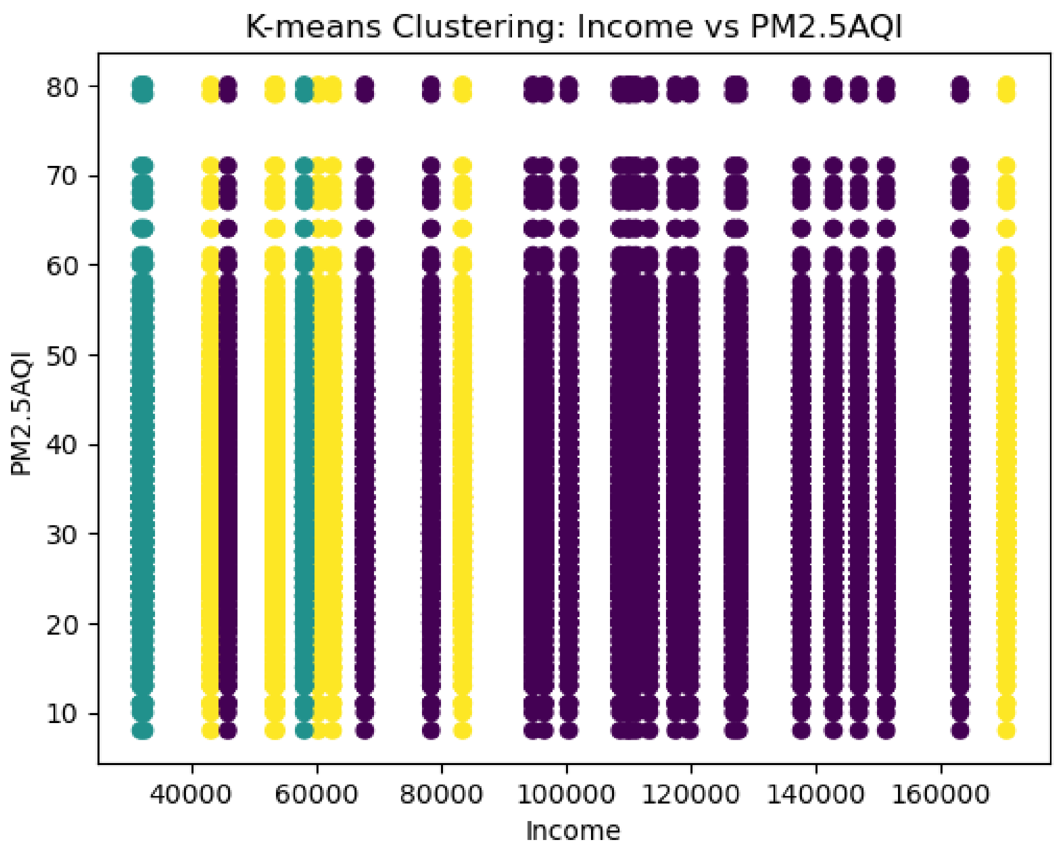


[ ]:
```python
#Jonathan
```

[6]:
```python
# Comparing the relationship between Median Income to PM2.5 Air Quality per Zipcode

import pandas as pd
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from sklearn.impute import SimpleImputer

# Load data
air_quality_data = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/ds-boston-transit-air-quality/fa23-team-b/data/AQI_30_zipcodes.csv')
income_data = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/ds-boston-transit-air-quality/fa23-team-b/data/Zipcodes_with_Median_Income.csv')

# Merge data on zip code, keeping only the rows where zip codes match
merged_data = pd.merge(air_quality_data, income_data, how='inner', left_on='zip_code', right_on='Zipcode')
```

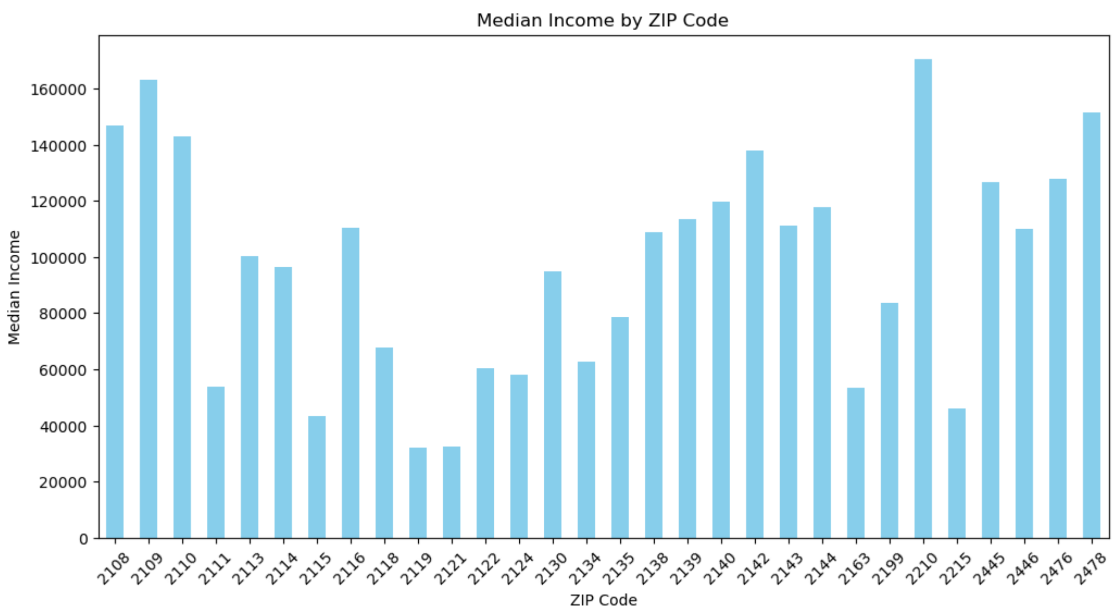
```
# Handle missing values
imputer = SimpleImputer(strategy='mean')
merged_data[['OZONEAQI', 'PM2.5AQI', 'Income_HH_Median', 'Asian', 'Black',␣
 ↪'White']] = imputer.fit_transform(merged_data[['OZONEAQI', 'PM2.5AQI',␣
 ↪'Income_HH_Median', 'Asian', 'Black', 'White']])

# Select relevant columns for analysis
features = merged_data[['OZONEAQI', 'PM2.5AQI', 'Income_HH_Median', 'Asian',␣
 ↪'Black', 'White']]

# Standardize the features
features = (features - features.mean()) / features.std()

# Perform K-means clustering
kmeans = KMeans(n_clusters=3, random_state=42)
merged_data['cluster'] = kmeans.fit_predict(features)

# Visualize the clusters
plt.scatter(merged_data['Income_HH_Median'], merged_data['PM2.5AQI'],␣
 ↪c=merged_data['cluster'], cmap='viridis')
plt.xlabel('Income')
plt.ylabel('PM2.5AQI')
plt.title('K-means Clustering: Income vs PM2.5AQI')
plt.show()
```

```
/Users/jonathansuarez/anaconda3/lib/python3.11/site-
packages/sklearn/cluster/_kmeans.py:1416: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init`
explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



[8]:
```
# Displaying the Median Income per Zipcode

import pandas as pd
import matplotlib.pyplot as plt

# Read the first CSV file
air_quality_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/AQI_30_zipcodes.csv')

# Read the second CSV file
income_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/Zipcodes_with_Median_Income.
↪csv')

# Merge Data
merged_df = pd.merge(air_quality_df, income_df, left_on='zip_code',␣
↪right_on='Zipcode', how='inner')

# Define Income Ranges and Labels
income_ranges = [0, 50000, 100000, 150000, 200000, float('inf')]
```

```
income_labels = ['0-50k', '50k-100k', '100k-150k', '150k-200k', '200k+']

# Categorize Data into Income Ranges
merged_df['IncomeRange'] = pd.cut(merged_df['Income_HH_Median'],␣
↪bins=income_ranges, labels=income_labels)

# Plotting
# Calculate Median Income for Each ZIP Code
median_income_by_zip = merged_df.groupby('zip_code')['Income_HH_Median'].
↪median()

# Plotting
plt.figure(figsize=(12, 6))
median_income_by_zip.plot(kind='bar', color='skyblue')
plt.title('Median Income by ZIP Code')
plt.xlabel('ZIP Code')
plt.ylabel('Median Income')
plt.xticks(rotation=45)
plt.show()
```



[ ]:
```
#Naveen
```

[10]:
```
from census import Census
from us import states

c = Census("e5097e26b8466d7dda3a7dec5b716652139fc17d", year=2020)
```

```
def get_population_density(zipcodes):
    density_data = []

    for zipcode in zipcodes:
        population_result = c.acs5dp.get(('NAME', 'DP05_0001E'), {'for': f'zip code tabulation area:{zipcode}'})
        land_area_result = c.acs5dp.get(('NAME', 'DP02_0010E'), {'for': f'zip code tabulation area:{zipcode}'})

        if population_result and land_area_result:
            population = population_result[0]['DP05_0001E']
            land_area = land_area_result[0]['DP02_0010E']

            if land_area > 0:
                # Convert land area from square meters to square miles
                land_area_sq_miles = land_area / 2.58999e6
                population_density = population / land_area_sq_miles
                density_data.append({'zipcode': zipcode, 'population_density': population_density})

    return density_data

zipcodes_to_query = ["02108", "02109", "02110", "02111", "02113", "02114", "02115", "02116", "02118", "02143",
    "02478", "02445", "02446", "02476", "02142", "02163", "02144", "02199", "02210", "02215",
    "02119", "02120", "02121", "02122", "02124", "02130", "02134", "02138", "02139", "02140"]  # Add your list of ZIP codes here

density_data = get_population_density(zipcodes_to_query)

for data in density_data:
    print(f"ZIP Code: {data['zipcode']}, Population Density: {data['population_density']:.2f} persons per square mile")

c = Census("e5097e26b8466d7dda3a7dec5b716652139fc17d", year=2020)

def get_housing_density(zipcodes):
    density_data = []

    for zipcode in zipcodes:
        housing_units_result = c.acs5dp.get(('NAME', 'DP04_0088E'), {'for': f'zip code tabulation area:{zipcode}'})
        land_area_result = c.acs5dp.get(('NAME', 'DP02_0010E'), {'for': f'zip code tabulation area:{zipcode}'})
```

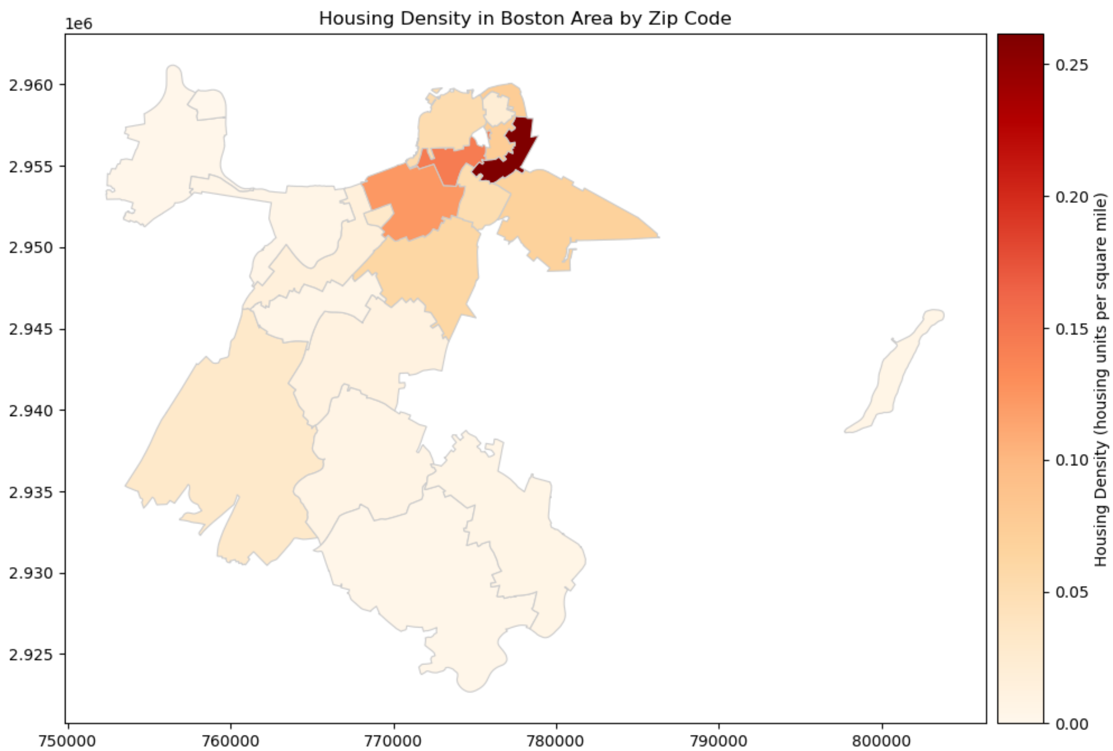
```
        if housing_units_result and land_area_result:
            housing_units = housing_units_result[0]['DP04_0088E']
            land_area = land_area_result[0]['DP02_0010E']

            if land_area > 0:
                # Convert land area from square meters to square miles
                land_area_sq_miles = land_area / 2.58999e6
                housing_density = housing_units / land_area_sq_miles
                density_data.append({'zipcode': zipcode, 'housing_density':␣
↪housing_density})

    return density_data

zipcodes_to_query = ["02108", "02109", "02110", "02111", "02113", "02114",␣
↪"02115", "02116", "02118", "02143",
    "02478", "02445", "02446", "02476", "02142", "02163", "02144", "02199",␣
↪"02210", "02215",
    "02119", "02120", "02121", "02122", "02124", "02130", "02134", "02138",␣
↪"02139", "02140"]  # Add your list of ZIP codes here

density_data = get_housing_density(zipcodes_to_query)

for data in density_data:
    print(f"ZIP Code: {data['zipcode']}, Housing Density:␣
↪{data['housing_density']:.2f} housing units per square mile")
```

```
ZIP Code: 02108, Population Density: 21559401.10 persons per square mile
ZIP Code: 02109, Population Density: 13483510.17 persons per square mile
ZIP Code: 02110, Population Density: 13900405.05 persons per square mile
ZIP Code: 02111, Population Density: 15479571.81 persons per square mile
ZIP Code: 02113, Population Density: 10524881.84 persons per square mile
ZIP Code: 02114, Population Density: 12964615.86 persons per square mile
ZIP Code: 02115, Population Density: 16968016.34 persons per square mile
ZIP Code: 02116, Population Density: 14245254.59 persons per square mile
ZIP Code: 02118, Population Density: 14524454.79 persons per square mile
ZIP Code: 02143, Population Density: 18517211.33 persons per square mile
ZIP Code: 02478, Population Density: 28249815.77 persons per square mile
ZIP Code: 02445, Population Density: 17756964.52 persons per square mile
ZIP Code: 02446, Population Density: 15298447.89 persons per square mile
ZIP Code: 02476, Population Density: 19291187.73 persons per square mile
ZIP Code: 02142, Population Density: 17763668.79 persons per square mile
ZIP Code: 02163, Population Density: 28489890.00 persons per square mile
ZIP Code: 02144, Population Density: 18677352.92 persons per square mile
ZIP Code: 02199, Population Density: 9385443.56 persons per square mile
ZIP Code: 02210, Population Density: 17083393.34 persons per square mile
ZIP Code: 02215, Population Density: 21632433.98 persons per square mile
ZIP Code: 02119, Population Density: 11742943.58 persons per square mile
```

```
ZIP Code: 02120, Population Density: 15987722.36 persons per square mile
ZIP Code: 02121, Population Density: 13821693.61 persons per square mile
ZIP Code: 02122, Population Density: 21845849.87 persons per square mile
ZIP Code: 02124, Population Density: 15414204.47 persons per square mile
ZIP Code: 02130, Population Density: 16073544.50 persons per square mile
ZIP Code: 02134, Population Density: 16924961.39 persons per square mile
ZIP Code: 02138, Population Density: 23017624.57 persons per square mile
ZIP Code: 02139, Population Density: 23034897.62 persons per square mile
ZIP Code: 02140, Population Density: 18465685.76 persons per square mile
ZIP Code: 02108, Housing Density: 3114665.69 housing units per square mile
ZIP Code: 02109, Housing Density: 1007835.88 housing units per square mile
ZIP Code: 02110, Housing Density: 3635490.55 housing units per square mile
ZIP Code: 02111, Housing Density: 794523.25 housing units per square mile
ZIP Code: 02113, Housing Density: 226588.27 housing units per square mile
ZIP Code: 02114, Housing Density: 683428.84 housing units per square mile
ZIP Code: 02115, Housing Density: 262668.20 housing units per square mile
ZIP Code: 02116, Housing Density: 1751633.21 housing units per square mile
ZIP Code: 02118, Housing Density: 877238.46 housing units per square mile
ZIP Code: 02143, Housing Density: 449984.02 housing units per square mile
ZIP Code: 02478, Housing Density: 2677584.65 housing units per square mile
ZIP Code: 02445, Housing Density: 2185006.60 housing units per square mile
ZIP Code: 02446, Housing Density: 1161118.78 housing units per square mile
ZIP Code: 02476, Housing Density: 653826.63 housing units per square mile
ZIP Code: 02142, Housing Density: 575553.33 housing units per square mile
ZIP Code: 02163, Housing Density: 0.00 housing units per square mile
ZIP Code: 02144, Housing Density: 699028.44 housing units per square mile
ZIP Code: 02199, Housing Density: 294317.05 housing units per square mile
ZIP Code: 02210, Housing Density: 1151943.23 housing units per square mile
ZIP Code: 02215, Housing Density: 93147.32 housing units per square mile
ZIP Code: 02119, Housing Density: 142580.04 housing units per square mile
ZIP Code: 02120, Housing Density: 76732.66 housing units per square mile
ZIP Code: 02121, Housing Density: 87408.07 housing units per square mile
ZIP Code: 02122, Housing Density: 124712.98 housing units per square mile
ZIP Code: 02124, Housing Density: 59629.94 housing units per square mile
ZIP Code: 02130, Housing Density: 504533.57 housing units per square mile
ZIP Code: 02134, Housing Density: 45878.83 housing units per square mile
ZIP Code: 02138, Housing Density: 1390855.21 housing units per square mile
ZIP Code: 02139, Housing Density: 1221890.44 housing units per square mile
ZIP Code: 02140, Housing Density: 800173.84 housing units per square mile
```

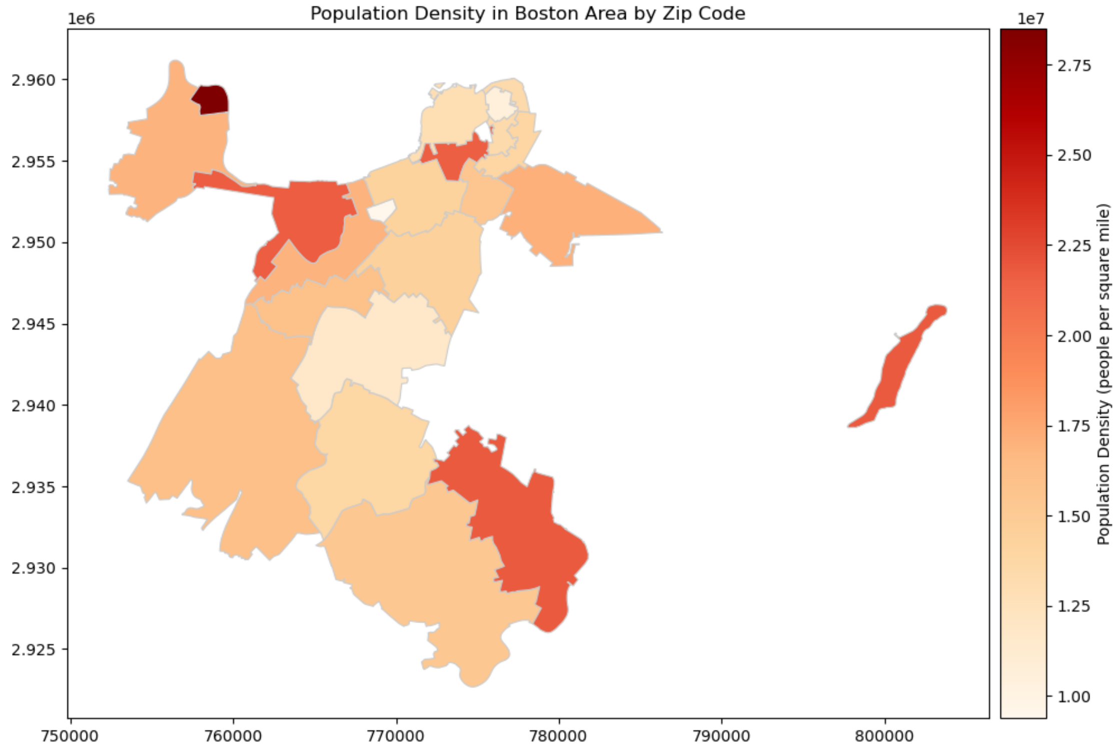
[21]:
```python
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable
from IPython.display import Image
```

```
data = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/density_data.csv')

data['zipcode'] = data['zipcode'].astype(str).str.zfill(5)

boston_zip_shapefile = gpd.read_file('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/ZIP_Codes.shp')

boston_zip_shapefile['ZIP5'] = boston_zip_shapefile['ZIP5'].astype(str).str.
↪zfill(5)

merged_data = boston_zip_shapefile.merge(data, left_on='ZIP5',␣
↪right_on='zipcode')

fig, ax = plt.subplots(1, 1, figsize=(12, 8))
divider = make_axes_locatable(ax)
cax = divider.append_axes("right", size="5%", pad=0.1)

merged_data.plot(column='housing_density', cmap='OrRd', linewidth=0.8, ax=ax,␣
↪edgecolor='0.8', legend=True, cax=cax,
                legend_kwds={'label': "Housing Density (housing units per␣
↪square mile)"})
ax.set_title('Housing Density in Boston Area by Zip Code')

plt.show()
```



[23]:

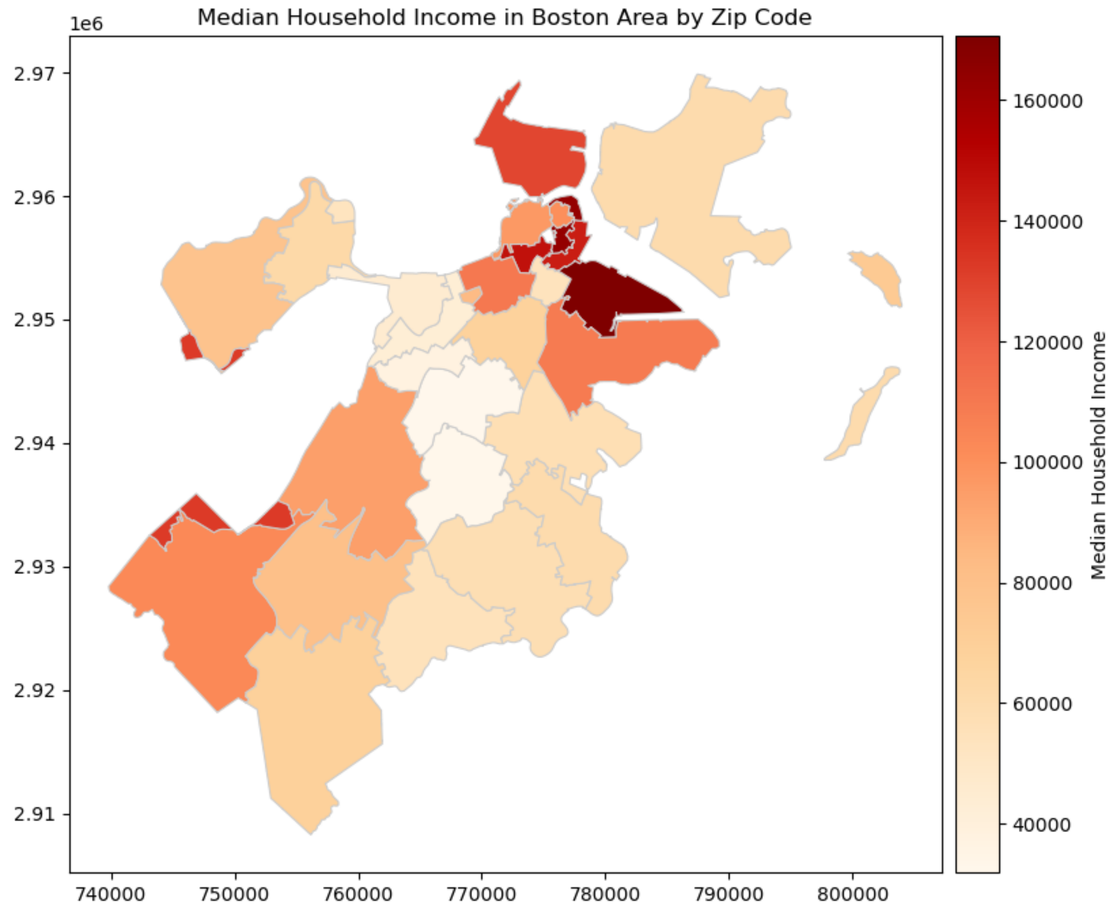
```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable
from IPython.display import Image

data = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
 ↪ds-boston-transit-air-quality/fa23-team-b/data/density_data.csv')

data['zipcode'] = data['zipcode'].astype(str).str.zfill(5)

boston_zip_shapefile = gpd.read_file('/Users/jonathansuarez/Documents/GitHub/
 ↪ds-boston-transit-air-quality/fa23-team-b/data/ZIP_Codes.shp')

boston_zip_shapefile['ZIP5'] = boston_zip_shapefile['ZIP5'].astype(str).str.
 ↪zfill(5)

merged_data = boston_zip_shapefile.merge(data, left_on='ZIP5',␣
 ↪right_on='zipcode')

fig, ax = plt.subplots(1, 1, figsize=(12, 8))
divider = make_axes_locatable(ax)
cax = divider.append_axes("right", size="5%", pad=0.1)

merged_data.plot(column='population_density', cmap='OrRd', linewidth=0.8,␣
 ↪ax=ax, edgecolor='0.8', legend=True, cax=cax,
                legend_kwds={'label': "Population Density (people per square␣
 ↪mile)"})
ax.set_title('Population Density in Boston Area by Zip Code')

plt.show()
```



[24]:

```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable

data = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
 ↪ds-boston-transit-air-quality/fa23-team-b/data/Zipcodes_with_Median_Income.
 ↪csv')

data['Zipcode'] = data['Zipcode'].astype(str).str.zfill(5)

boston_zip_shapefile = gpd.read_file('/Users/jonathansuarez/Documents/GitHub/
 ↪ds-boston-transit-air-quality/fa23-team-b/data/ZIP_Codes.shp')

boston_zip_shapefile['ZIP5'] = boston_zip_shapefile['ZIP5'].astype(str).str.
 ↪zfill(5)

merged_data = boston_zip_shapefile.merge(data, left_on='ZIP5',␣
 ↪right_on='Zipcode')

fig, ax = plt.subplots(1, 1, figsize=(12, 8))
divider = make_axes_locatable(ax)
cax = divider.append_axes("right", size="5%", pad=0.1)
```

```
merged_data.plot(column='Income_HH_Median', cmap='OrRd', linewidth=0.8, ax=ax,␣
↪edgecolor='0.8', legend=True, cax=cax,
                legend_kwds={'label': "Median Household Income"})
ax.set_title('Median Household Income in Boston Area by Zip Code')

plt.show()
```



[ ]:
```
#Chao-Jen
```

[14]:
```
import pandas as pd

aqi_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/AQI_30_zipcodes.csv')

density_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/density_data.csv')

aqi_df['date'] = pd.to_datetime(aqi_df['date'])
```

```python
aqi_df['year'] = aqi_df['date'].dt.year

result = aqi_df.groupby(['zip_code', 'year'])[['OZONEAQI', 'PM2.5AQI']].mean()

result = result.reset_index()

result = result.merge(density_df, left_on='zip_code', right_on='zipcode',␣
 ↪how='left')

result = result.drop(columns=['zipcode'])

result['population_density'] = result['population_density'].apply(lambda x: "{:.
 ↪1f}".format(x))

print(result)
```

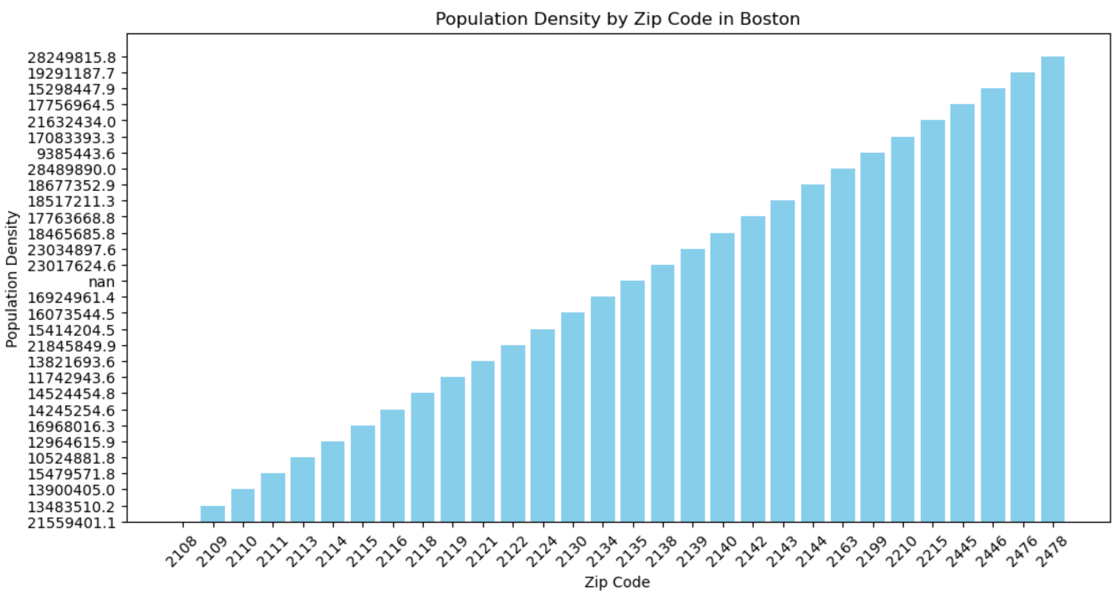
```
    zip_code  year   OZONEAQI   PM2.5AQI  population  housing_density  \
0       2108  2020  30.756906  34.352459      4520.0         0.144469
1       2109  2020  30.756906  34.352459      3639.0         0.074746
2       2110  2020  30.756906  34.352459      2340.0         0.261538
3       2111  2020  30.756906  34.352459      7949.0         0.051327
4       2113  2020  30.756906  34.352459      7339.0         0.021529
5       2114  2020  30.756906  34.352459     13260.0         0.052715
6       2115  2020  30.756906  34.352459     29134.0         0.015480
7       2116  2020  30.756906  34.352459     23007.0         0.122963
8       2118  2020  30.756906  34.352459     28892.0         0.060397
9       2119  2020  30.756906  34.352459     27426.0         0.012142
10      2121  2020  30.756906  34.352459     29570.0         0.006324
11      2122  2020  30.756906  34.352459     24874.0         0.005709
12      2124  2020  30.756906  34.352459     57128.0         0.003869
13      2130  2020  30.756906  34.352459     42021.0         0.031389
14      2134  2020  30.756906  34.352459     19552.0         0.002711
15      2135  2020  30.756906  34.352459         NaN              NaN
16      2138  2020  30.756906  34.352459     39139.0         0.060426
17      2139  2020  30.756906  34.352459     39702.0         0.053045
18      2140  2020  30.756906  34.352459     20954.0         0.043333
19      2142  2020  30.756906  34.352459      4074.0         0.032401
20      2143  2020  30.756906  34.352459     25102.0         0.024301
21      2144  2020  30.756906  34.352459     25009.0         0.037427
22      2163  2020  30.756906  34.352459      2343.0         0.000000
23      2199  2020  30.756906  34.352459      1435.0         0.031359
24      2210  2020  30.756906  34.352459      4538.0         0.067431
25      2215  2020  30.756906  34.352459     26243.0         0.004306
26      2445  2020  30.756906  34.352459     20520.0         0.123051
27      2446  2020  30.756906  34.352459     29711.0         0.075898
28      2476  2020  30.756906  34.352459     17526.0         0.033893
```

```
29      2478  2020  30.756906  34.352459     26123.0         0.094782

   population_density
0          21559401.1
1          13483510.2
2          13900405.0
3          15479571.8
4          10524881.8
5          12964615.9
6          16968016.3
7          14245254.6
8          14524454.8
9          11742943.6
10         13821693.6
11         21845849.9
12         15414204.5
13         16073544.5
14         16924961.4
15                nan
16         23017624.6
17         23034897.6
18         18465685.8
19         17763668.8
20         18517211.3
21         18677352.9
22         28489890.0
23          9385443.6
24         17083393.3
25         21632434.0
26         17756964.5
27         15298447.9
28         19291187.7
29         28249815.8
```

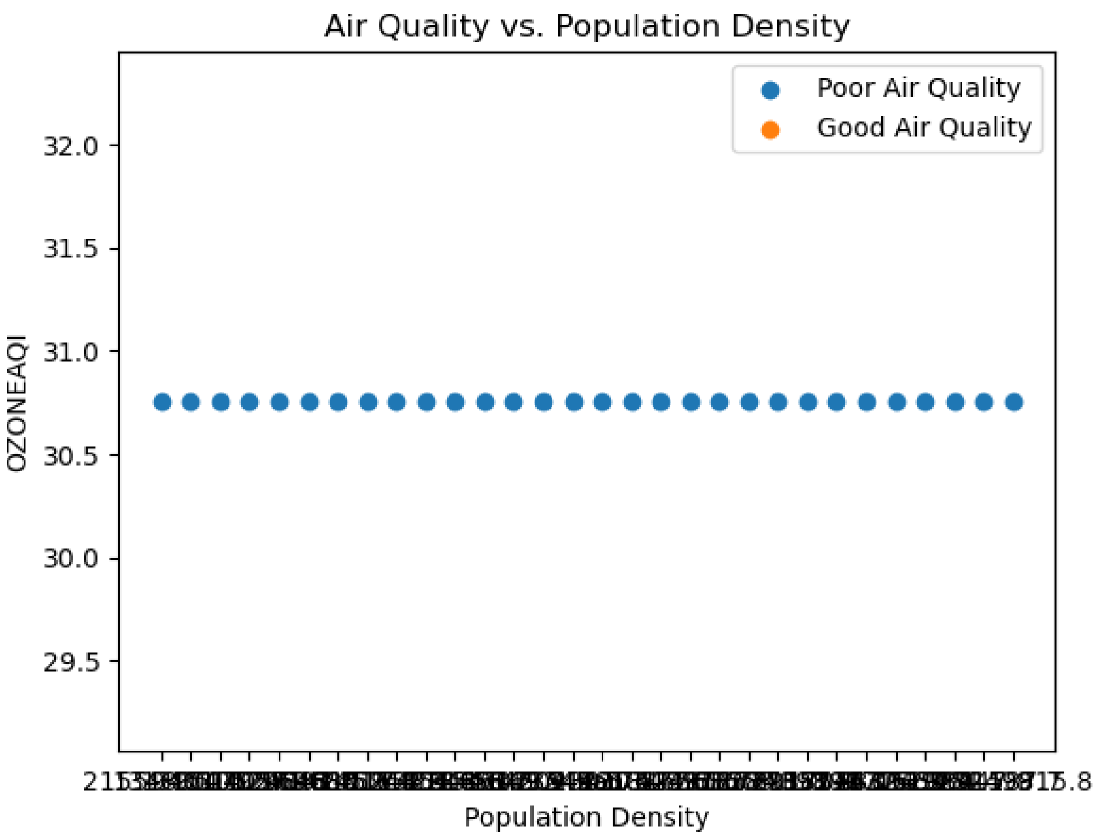
[15]:

```python
import matplotlib.pyplot as plt

zip_codes = result['zip_code']
population_density = result['population_density']

plt.figure(figsize=(12, 6))
plt.bar(range(len(zip_codes)), population_density, color='skyblue')

plt.xticks(range(len(zip_codes)), zip_codes, rotation=45)

plt.xlabel('Zip Code')
plt.ylabel('Population Density')
plt.title('Population Density by Zip Code in Boston')
```

```
plt.show()
```



[16]:
```
import matplotlib.pyplot as plt


poor_air_quality = result[result['OZONEAQI'] > 30]


good_air_quality = result[result['OZONEAQI'] <= 30]


plt.scatter(poor_air_quality['population_density'], poor_air_quality['OZONEAQI'], label='Poor Air Quality')
plt.scatter(good_air_quality['population_density'], good_air_quality['OZONEAQI'], label='Good Air Quality')

plt.xlabel('Population Density')
plt.ylabel('OZONEAQI')
plt.legend()
plt.title('Air Quality vs. Population Density')
plt.show()
```



[17]:
```
result['population_density'] = pd.to_numeric(result['population_density'],␣
 ↪errors='coerce')


correlation_ozone = result['population_density'].corr(result['OZONEAQI'],␣
 ↪method='pearson')
correlation_pm25 = result['population_density'].corr(result['PM2.5AQI'],␣
 ↪method='pearson')

print(f'Correlation between Population Density and OZONEAQI: {correlation_ozone:
 ↪.2f}')
print(f'Correlation between Population Density and PM2.5AQI: {correlation_pm25:.
 ↪2f}')
```

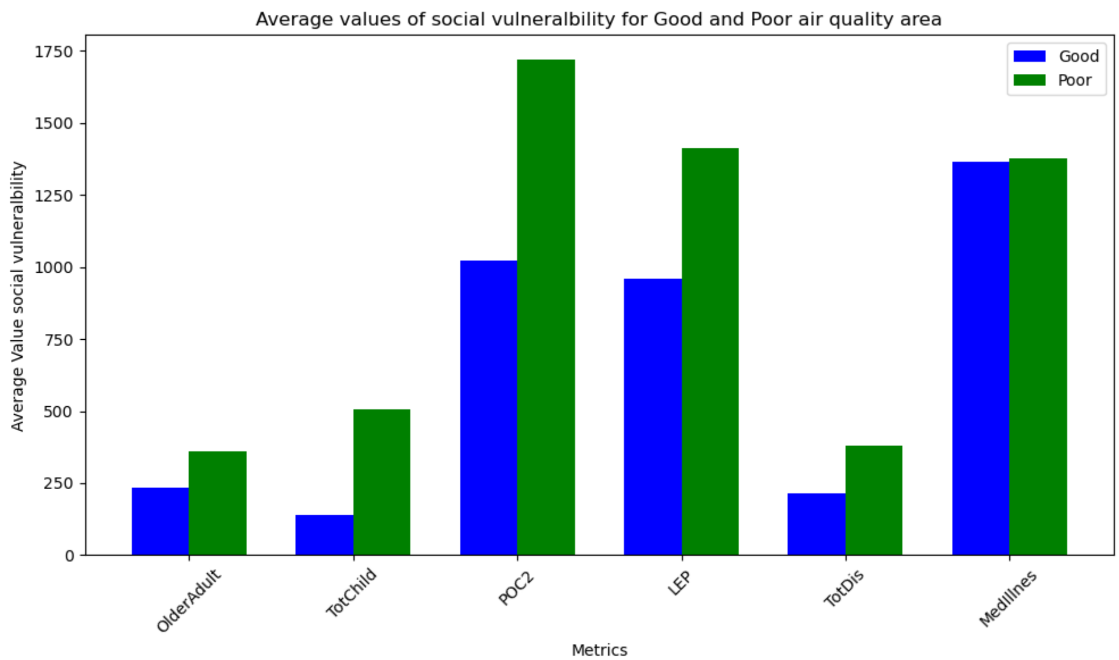
```
Correlation between Population Density and OZONEAQI: 0.00
Correlation between Population Density and PM2.5AQI: -0.00
```

[ ]:
```
#Cathy
```

[18]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Assuming you have the data extracted from the images into two csv files:␣
↪'social_vulnerability.csv' and 'air_quality.csv'

social_vulnerability_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/
↪Climate_Ready_Boston_Social_Vulnerability.csv')
aqi_df = pd.read_csv('/Users/jonathansuarez/Documents/GitHub/
↪ds-boston-transit-air-quality/fa23-team-b/data/AQI_30_zipcodes.csv')
selected_names = ['Fenway', 'Back Bay', 'Allston', 'Brighton', 'Dorchester',
                  'North End', 'West End', 'Longwood Medical Area', 'South End',
                  'Bay Village', 'Leather District', 'Harbor Islands']

filtered_df = social_vulnerability_df[social_vulnerability_df['Name'].
↪isin(selected_names)].copy()

selected_zip_codes = [2115, 2116, 2134, 2135, 2124, 2114, 2113, 2118, 2111,␣
↪2110]

filtered_aqi_df = aqi_df[aqi_df['zip_code'].isin(selected_zip_codes)]

name_to_zip = {
    'Fenway': 2115,
    'Back Bay' : 2116,
    'Allston': 2134,
    'Brighton': 2135,
    'Dorchester': 2124,
    'North End': 2114,
    'West End': 2113,
    'Longwood Medical Area': 2115,
    'South End': 2118,
    'Bay Village': 2116,
    'Leather District': 2111,
    'Harbor Islands': 2110
}

filtered_df.loc[:, 'zip_code'] = filtered_df['Name'].map(name_to_zip)

merged_df = pd.merge(filtered_aqi_df, filtered_df, on='zip_code', how='inner')
count_df = filtered_aqi_df[filtered_aqi_df['CategoryName'].isin(['Good',␣
↪'Moderate'])].groupby('zip_code')['CategoryName'].value_counts().unstack().
↪fillna(0)
count_df['Total_Good'] = count_df['Good']
```

```python
sorted_df = count_df.sort_values(by='Total_Good', ascending=False)
top_5_zip = sorted_df.head(5).index.tolist()
bottom_5_zip = sorted_df.tail(5).index.tolist()
sorted_df['AirQuality'] = 'neutral'
sorted_df.loc[top_5_zip, 'AirQuality'] = 'good'
sorted_df.loc[bottom_5_zip, 'AirQuality'] = 'poor'
aqi_df = aqi_df.merge(sorted_df['AirQuality'], on='zip_code', how='left')

sorted_df = sorted_df.reset_index()

merged_df = filtered_df.merge(sorted_df[['zip_code', 'AirQuality']],␣
  ↪on='zip_code', how='inner')

good_data = merged_df[merged_df['AirQuality'] == 'good']
poor_data = merged_df[merged_df['AirQuality'] == 'poor']

metrics = ['OlderAdult', 'TotChild', 'POC2', 'LEP', 'TotDis', 'MedIllnes']

good_averages = good_data[metrics].mean()
poor_averages = poor_data[metrics].mean()

bar_width = 0.35
index = np.arange(len(metrics))

fig, ax = plt.subplots(figsize=(10, 6))
bar1 = ax.bar(index, good_averages, bar_width, label='Good', color='b')
bar2 = ax.bar(index + bar_width, poor_averages, bar_width, label='Poor',␣
  ↪color='g')

ax.set_xlabel('Metrics')
ax.set_ylabel('Average Value social vulneralbility')
ax.set_title('Average values of social vulneralbility for Good and Poor air␣
  ↪quality area')
ax.set_xticks(index + bar_width / 2)
ax.set_xticklabels(metrics, rotation=45)
ax.legend()

plt.tight_layout()
plt.show()
```



BASE QUESTION NUMBER 2:

Analyze yearly health data and proximity to diverse transportation infrastructure (e.g. public transit and roads) to investigate the relationship between these data and the disparate impacts on the residents of Boston. This means answering the following questions:

How do areas with poor air quality compare to areas with better air quality based on different demographic characteristics, specifically:

Race/ethnicity (ACS)?

With regards to race and ethnicity, we can see from the PPI data that as air quality decreases, the proportion of minorities living in these areas increases. While the population density of non-Hispanic Whites remains the dominant racial group throughout all of the PPI levels, the data shows a drastic increase in the number of all other racial categories particularly between levels 3 and 4, and the numbers increase even further between levels 4 and 5.

Area median income/ income?

We were able to demonstrate the median income by zip code in Boston to get a better understanding of which areas are considered wealthier and other areas poorer. As we can see with the K-means clustering graph, It doesn't seem that the air quality index differs as much by zipcode, as it is mostly clustered in the 20 to 60 range. This could be due to the fact that 2020 is when the Covid Pandemic happened, and transit activity was as its lowest it has ever been in the history of the United States. However, there was still typical transit activity in the months January to March, so there is opportunity to perform more in-depth analysis on the true Air Quality for those months.

Housing density?

As demonstrated through our shaded map analysis, we've uncovered interesting trends in Boston's housing density. The northeastern region of Boston stands out with the highest housing density, indicating a greater concentration of houses per square mile. In contrast, as we move south and west, the housing density decreases, suggesting less compact living conditions in those areas. What's particularly noteworthy is the correlation between the darker-shaded regions on the housing density map and some of the more expensive areas on the median household income graph. This correlation suggests that areas with higher housing density often coincide with higher-income neighborhoods in Boston The implications of this correlation are significant, indicating a potential connection between housing density and income levels in Boston. It raises the possibility that higher housing density areas might be more appealing to individuals with higher incomes, potentially due to factors such as proximity to urban amenities and employment opportunities. This finding holds considerable weight for urban planning and housing policy decisions, especially concerning air quality and transit. To provide a seamless transition to our conclusion, it's crucial to recognize that understanding the relationship between housing density and income can significantly inform decisions related to affordable, healthy urban development, as well as investments in infrastructure. Such decisions can collectively contribute to creating more equitable living conditions for all residents of Boston, ultimately fostering a vibrant and inclusive urban landscape."

Population density?

We can use zip codes to represent areas with higher and lower population densities. Next, we analyze the data in conjunction with the AQI data, creating a scatter plot to determine the correlation between the two. Based on the current data, correlation between Population Density and OZONEAQI/PM2.5AQI : 0.

Social vulnerability?

Regarding the social vulnerability comparison between good air quality area and poor air quality area, we compared them on the following metrics: older adults(OlderAdult), children(TotChild), people of color(POC2), limited English proficiency(LEP), lower income(Low_to_No), people of disabilities(TotDis), and medical illness(MedIllness). We analyzed the zipcodes based on the overlap of the zipcodes in the social vulnerability dataset that are assigned by the place names and the zipcodes present in our air quality dataset, which is a total of 10. First, we counted the zipcodes according to their CategoryName (which includes whether the daily air quality is Good or Moderate), and based on the ratio, we classified the 5 zipcodes with the most Good as good air quality areas, and the 5 zipcodes with the least Good as poor air quality areas. Based on this, we calculated an average for each metrics for the good and poor groups and displayed the result in a bar chart. We can see that in each of the metrics comparisons, the data for the POOR area is much higher than the GOOD area. This shows that these air quality has an impact on social vulnerability metrics. There is a strong correlation between the presence of poor air quality and people such as children, elderly, and low income adults who live in those areas.