Worst Quality of Life

Aidan McNulty, Brandon Allan, Liam Kren

Introduction

Our projects goal is to collectively find the “worst” place to live in Pittsburgh. In our case more so it’s what place(s) in Pittsburgh we would like to stay clear of. We knew that many people often look for the best place to live, but by finding the worst, it allows you to settle, even if you cannot get the best. Each of us found a dataset that we believe contributes to this research in its own way, and used what we found from each of the data to determine which is the worst place to live in based on these three factors.

The metric was easy to decide on because we each found datasets that were clear deciding factors of places that you would want to stay away from. We focused on three areas of importance, which were the Allegheny County Fatal Accidental Overdoses, Police incident reports, and firearm seizures in Pittsburgh. For each of these datasets that our group researched, we were able to narrow down the information to neighborhoods and general areas in which our dataset was most prevalent/had the most incidents. Using this newfound data, we compared our data using different charts to find out which place or area has the most incidents all together.

Areas of importance :

- Allegheny County Fatal Accidental Overdoses
- Police Incident Reports
- Firearm seizures in Pittsburgh

Allegheny County Fatal Accidental Overdoses

The criteria used to find the “worst” place to live from the CSV was based on the highest number of fatal overdoses attributed to a zip code.

However the base CSV file doesn’t contain an overall count.

```
import pandas as pd

data = pd.read_csv('data.csv')

# Display the first few rows to verify the data was loaded correctly
print(data.head())
```

```
       _id  death_date_and_time manner_of_death  age sex race case_dispo  \
0  9368074  2007-02-08T14:55:00        Accident   29   M    W         MO
1  9368075  2007-02-07T15:07:00        Accident   45   M    W         MO
2  9368076  2007-02-28T12:15:00        Accident   60   F    W         MO
3  9368077  2007-03-28T13:40:00        Accident   53   M    B         MO
4  9368078  2007-04-23T23:59:00        Accident   36   M    W         MO

  combined_od1 combined_od2 combined_od3 combined_od4 combined_od5  \
0      Cocaine     Fentanyl     Morphine    Oxycodone          NaN
1     Diazepam  Hydrocodone  Mirtazapine    Oxycodone    Trazodone
2      Cocaine       Heroin          NaN          NaN          NaN
3      Cocaine          NaN          NaN          NaN          NaN
4      Alcohol   Alprazolam       Heroin          NaN          NaN

  combined_od6 combined_od7 combined_od8 combined_od9 combined_od10  \
0          NaN          NaN          NaN          NaN           NaN
1          NaN          NaN          NaN          NaN           NaN
2          NaN          NaN          NaN          NaN           NaN
3          NaN          NaN          NaN          NaN           NaN
4          NaN          NaN          NaN          NaN           NaN

  incident_zip  decedent_zip  case_year
0        15226           NaN       2007
1        15068           NaN       2007
2        15220           NaN       2007
3        15211           NaN       2007
4        15227           NaN       2007
```

Allegheny County Fatal Accidental Overdoses

By adding an extra column to our data we can see how many times a zip code occurred within the data set.

This way we will be able to create visualizations of the data later on.

```
       _id  death_date_and_time manner_of_death  age sex race case_dispo  \
0  9368074  2007-02-08T14:55:00        Accident   29   M    W         MO
1  9368075  2007-02-07T15:07:00        Accident   45   M    W         MO
2  9368076  2007-02-28T12:15:00        Accident   60   F    W         MO
3  9368077  2007-03-28T13:40:00        Accident   53   M    B         MO
4  9368078  2007-04-23T23:59:00        Accident   36   M    W         MO

  combined_od1 combined_od2 combined_od3  ... combined_od5 combined_od6  \
0      Cocaine     Fentanyl     Morphine  ...          NaN          NaN
1     Diazepam  Hydrocodone  Mirtazapine  ...     Trazodone         NaN
2      Cocaine       Heroin          NaN  ...          NaN          NaN
3      Cocaine          NaN          NaN  ...          NaN          NaN
4      Alcohol   Alprazolam       Heroin  ...          NaN          NaN

  combined_od7 combined_od8 combined_od9 combined_od10 incident_zip  \
0          NaN          NaN          NaN           NaN        15226
1          NaN          NaN          NaN           NaN        15068
2          NaN          NaN          NaN           NaN        15220
3          NaN          NaN          NaN           NaN        15211
4          NaN          NaN          NaN           NaN        15227

  decedent_zip  case_year  incident_count
0          NaN       2007           118.0
1          NaN       2007            50.0
2          NaN       2007            72.0
3          NaN       2007           121.0
4          NaN       2007           171.0
```

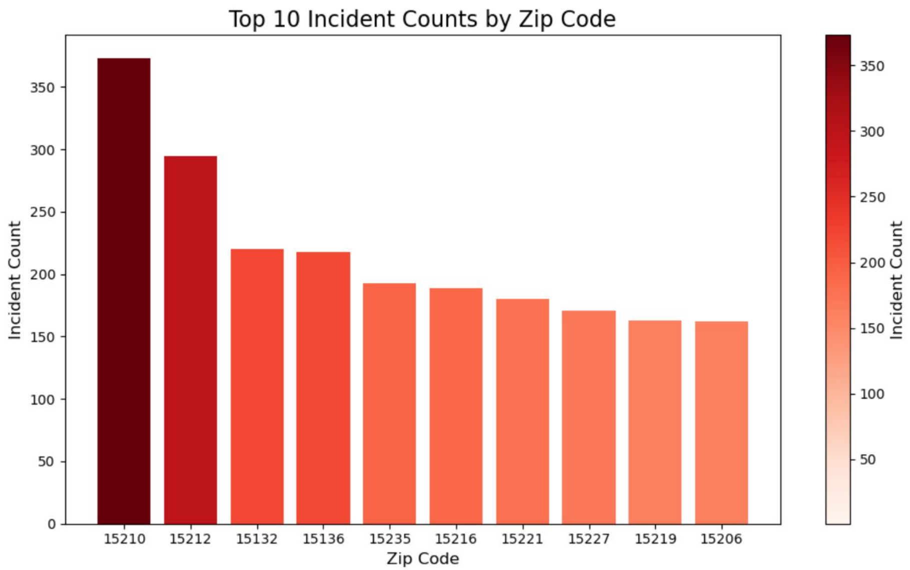
```python
# Group by the 'incident_zip' column and count occurrences
zip_counts = data['incident_zip'].value_counts().reset_index()
zip_counts.columns = ['incident_zip', 'incident_count']

# Merge the counts back with the original data on 'incident_zip'
data_with_counts = data.merge(zip_counts, on='incident_zip', how='left')

# Display the first few rows of the updated data
print(data_with_counts.head())
```

Allegheny County Fatal Accidental Overdoses

The next step for me was to create a visualization based on the new column.

For the first visualization I created a bar chart showing the **Top 10** incident counts by zip code.

I did this by using both pandas and matplotlib, sorting by the new incident count column.

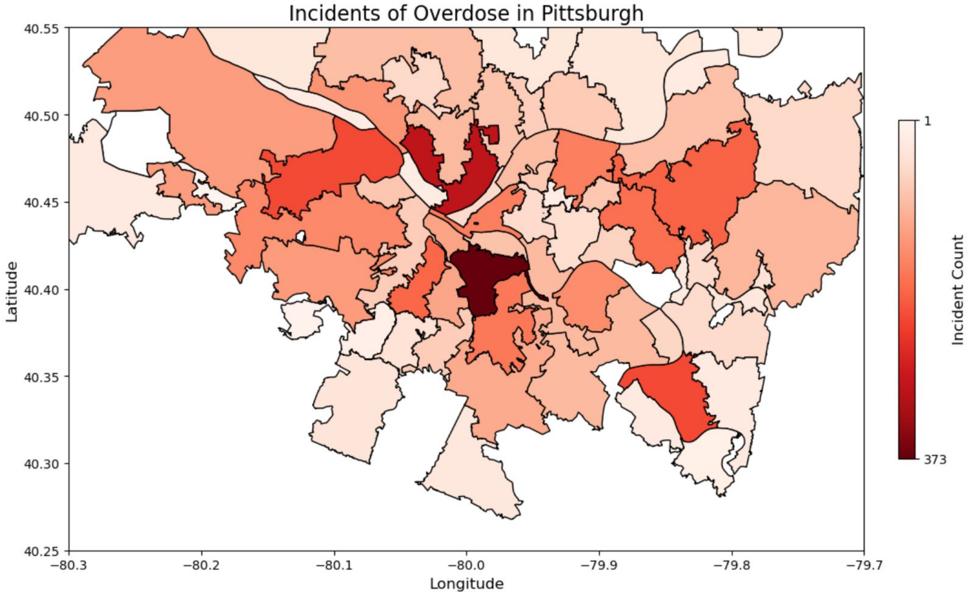


Allegheny County Fatal Accidental Overdoses

Finally I also wanted to have a heat map of only Pittsburgh's counties to see where the highest incidents are on a map.

In conclusion of my data we can see that the zip code **15210(Allentown)** contained the most deaths related to drug overdose.

This is tailed by **15212(Brighton Heights / Allegheny Center), 15132(McKeesport),** and **15136(McKees Rocks).**



```python
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.colors as mcolors

# Load the CSV data into a pandas DataFrame
data = pd.read_csv('data.csv')

# Group by the 'incident_zip' column and count occurrences
zip_counts = data['incident_zip'].value_counts().reset_index()
zip_counts.columns = ['incident_zip', 'incident_count']

# Merge the counts back with the original data on 'incident_zip'
data_with_counts = data.merge(zip_counts, on='incident_zip', how='left')

# Group by 'incident_zip' and get the maximum 'incident_count' for each zip code
zip_counts_sorted = data_with_counts.groupby('incident_zip')['incident_count'].max().sort_values(ascending=False)

# Select only the top 10 zip codes
top_10_zip_counts = zip_counts_sorted.head(10)

# Normalize the incident counts for color mapping (using linear normalization)
norm = mcolors.Normalize(vmin=zip_counts_sorted.min(), vmax=zip_counts_sorted.max())
cmap = plt.cm.Reds  # Red colormap (deep red for more incidents)

# Create a new column for color based on the incident count
colors = [cmap(norm(count)) for count in top_10_zip_counts]

# Plot the bar chart for top 10 zip codes
fig, ax = plt.subplots(figsize=(10, 6))
bars = ax.bar(top_10_zip_counts.index, top_10_zip_counts.values, color=colors)

# Add title and labels
ax.set_title('Top 10 Incident Counts by Zip Code', fontsize=16)
ax.set_xlabel('Zip Code', fontsize=12)
ax.set_ylabel('Incident Count', fontsize=12)

# Set fixed ticks and labels
ax.set_xticks(range(len(top_10_zip_counts)))  # Set tick positions
ax.set_xticklabels(top_10_zip_counts.index, rotation=0)  # Set labels horizontal (rotation=0)

# Add color bar
sm = plt.cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])  # Empty array because color bar needs scalar data
cbar = fig.colorbar(sm, ax=ax)  # Link the color bar to the axis
cbar.set_label('Incident Count', fontsize=12)

# Show the plot
plt.tight_layout()
plt.show()
```

```python
# Load the GeoJSON data from the URL
geojson_url = "https://data.wprdc.org/dataset/1a5135de-cabe-4e23-b5e4-b2b8dd733817/resource/14e5de97-0a5f-45
response = requests.get(geojson_url)

# Load the GeoJSON data into a GeoDataFrame
gdf = gpd.read_file(StringIO(response.text))

# Filter the GeoDataFrame to include only zip codes between 15106 and 15295
gdf_filtered = gdf[gdf['ZIP'].astype(str).between('15106', '15295')].copy()  # Make a copy to avoid the warn

# Merge the filtered GeoDataFrame with the incident counts for all zip codes in the range
gdf_filtered['incident_count'] = gdf_filtered['ZIP'].map(zip_counts_sorted)

# Normalize the incident counts for color mapping (using linear normalization)
norm = mcolors.Normalize(vmin=zip_counts_sorted.min(), vmax=zip_counts_sorted.max())
cmap = plt.cm.Reds  # Red colormap (deep red for more incidents)

# Create a new column for color based on the incident count
gdf_filtered['color'] = gdf_filtered['incident_count'].apply(
    lambda x: cmap(norm(x)) if pd.notna(x) else 'white'  # Set 'white' for NaN values (no incidents)
)

# Plot the data
fig, ax = plt.subplots(figsize=(12, 12))

# Plot the zip codes with the color mapped by incident counts
gdf_filtered.plot(ax=ax, edgecolor='black', color=gdf_filtered['color'])

# Set plot title and labels
ax.set_title("Incidents of Overdose in Pittsburgh", fontsize=16)
ax.set_xlabel("Longitude", fontsize=12)
ax.set_ylabel("Latitude", fontsize=12)

# Zoom out to show the specified range of zip codes
ax.set_xlim([-80.3, -79.7])  # Adjusted to zoom out slightly
ax.set_ylim([40.25, 40.55])  # Adjusted to zoom out slightly

# Add the color bar to the side, adjusting height to match the map
sm = plt.cm.ScalarMappable(cmap=cmap, norm=norm)
sm.set_array([])  # Empty array because color bar needs scalar data
cbar = fig.colorbar(sm, ax=ax, fraction=0.02, pad=0.04)  # Adjusted fraction and padding for size
cbar.set_label('Incident Count', fontsize=12)  # Label for the color bar
cbar.set_ticks([zip_counts_sorted.min(), zip_counts_sorted.max()])
cbar.ax.invert_yaxis()  # Color bar will now have high counts at the top

# Show the map with color bar
plt.show()
```

Police Incident Reports

A great metric to determine which neighborhood is the “worst” , is by looking at how dangerous each neighborhood is. By using public police incident reports, the most dangerous neighborhood can be found by counting the amount of individual incidents that occur in each neighborhood.

To count each time an incident occured in each neighborhood, code was used to create a table of the top ten neighborhoods with the most police incidents.

```
import pandas as pd
data = pd.read_csv("police.csv")
```

```
count = data['INCIDENTNEIGHBORHOOD'].value_counts()
```

```
top10 = count.head(10)
```

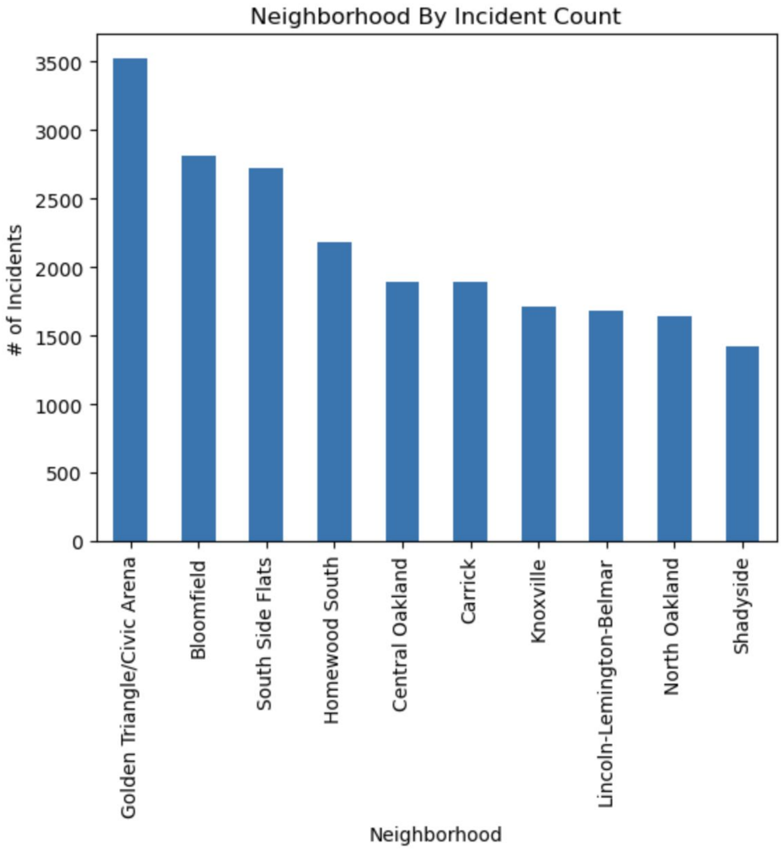
```
top10
```

```
INCIDENTNEIGHBORHOOD
Golden Triangle/Civic Arena     3524
Bloomfield                      2811
South Side Flats                2720
Homewood South                  2187
Central Oakland                 1891
Carrick                         1887
Knoxville                       1716
Lincoln-Lemington-Belmar        1680
North Oakland                   1644
Shadyside                       1419
```

```
top10.plot.bar(xlabel = "Neighborhood", ylabel = "# of Incidents", title = "Neighborhood By Incident Count")
```

Police Incident Reports

To show the newly found data in a more informative and visual way, a bar graph was made to display the data.

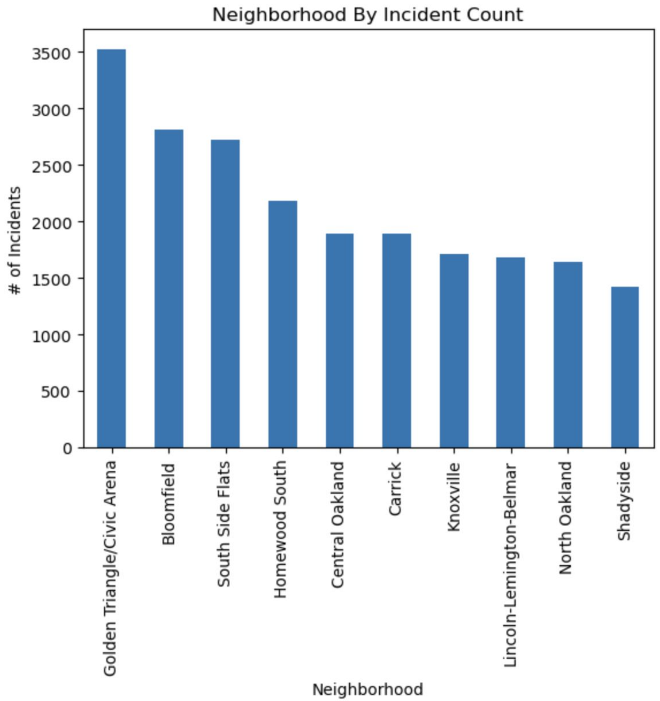


Police Incident Reports Conclusion

By using this data, it is shown that Golden Triangle/Civic Arena (Downtown) is the most dangerous neighborhood by far. It is followed by Bloomfield and South Side Flats. The top three most dangerous neighborhoods have 600+ more incidents than the following seven neighborhoods.

```
top10
```

```
INCIDENTNEIGHBORHOOD
Golden Triangle/Civic Arena    3524
Bloomfield                     2811
South Side Flats               2720
Homewood South                 2187
Central Oakland                1891
Carrick                        1887
Knoxville                      1716
Lincoln-Lemington-Belmar       1680
North Oakland                  1644
Shadyside                      1419
```



Firearm Seizures in Pittsburgh

One way that can be used to determine the “worst” place to live in Pittsburgh is by the amount of firearm seizures in an area. This is because it shows that the area has a past of many citizens owning either illegal firearms, or firearms that were used incorrectly/irresponsibly that got them taken away. This code is used to display 10 results of the data to show that it is imported, along with each category

```python
import pandas as pd

data_path = "firearms.csv"
df = pd.read_csv(data_path)
print(df.head(10))
```

```
   _id                                  address  total_count  \
0    1    1700 BLOCK ARLINGTON AVE PITTSBURGH, PA 15210            2
1    2      BREVET WAY & MINTON ST PITTSBURGH, PA 15204            1
2    3   TERRACE ST & WHITRIDGE ST PITTSBURGH, PA 15213            1
3    4        3500 BLOCK GERBER AVE PITTSBURGH, PA 15212            1
4    5          7500 BLOCK KELLY ST PITTSBURGH, PA 15208            1
5    6            1900 BLOCK 5TH AVE PITTSBURGH, PA 15219            1
6    7        400 BLOCK S NEGLEY AVE PITTSBURGH, PA 15232            1
7    8      5200 BLOCK DRESDEN WAY PITTSBURGH, PA 15201            2
8    9          1100 BLOCK GRAND AVE PITTSBURGH, PA 15212            1
9   10  1600 BLOCK FALLOWFIELD AVE PITTSBURGH, PA 15216            1

   other_count  pistol_count  revolver_count  rifle_count  shotgun_count  \
0            0             1               1            0              0
1            0             1               0            0              0
2            0             1               0            0              0
3            0             1               0            0              0
4            0             1               0            0              0
5            0             1               0            0              0
6            0             1               0            0              0
7            0             2               0            0              0
8            0             1               0            0              0
9            0             1               0            0              0
```


```
   _id                                  address  total_count  \
0    1    1700 BLOCK ARLINGTON AVE PITTSBURGH, PA 15210            2
1    2      BREVET WAY & MINTON ST PITTSBURGH, PA 15204            1
2    3   TERRACE ST & WHITRIDGE ST PITTSBURGH, PA 15213            1
3    4        3500 BLOCK GERBER AVE PITTSBURGH, PA 15212            1
4    5          7500 BLOCK KELLY ST PITTSBURGH, PA 15208            1
5    6            1900 BLOCK 5TH AVE PITTSBURGH, PA 15219            1
6    7        400 BLOCK S NEGLEY AVE PITTSBURGH, PA 15232            1
7    8      5200 BLOCK DRESDEN WAY PITTSBURGH, PA 15201            2
8    9          1100 BLOCK GRAND AVE PITTSBURGH, PA 15212            1
9   10  1600 BLOCK FALLOWFIELD AVE PITTSBURGH, PA 15216            1

   other_count  pistol_count  revolver_count  rifle_count  shotgun_count  \
0            0             1               1            0              0
1            0             1               0            0              0
2            0             1               0            0              0
3            0             1               0            0              0
4            0             1               0            0              0
5            0             1               0            0              0
6            0             1               0            0              0
7            0             2               0            0              0
8            0             1               0            0              0
9            0             1               0            0              0
```

Firearm Seizures in Pittsburgh

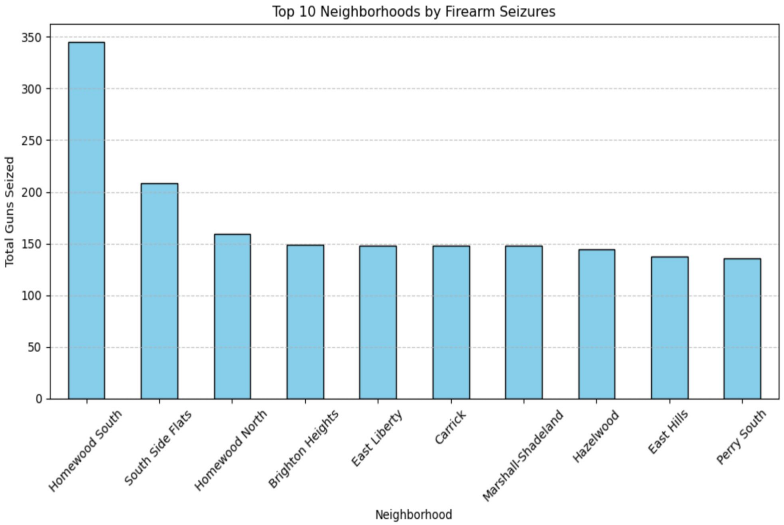
The data is first imported as a whole, which includes many unnecessary sections that are not needed for the topic that we are researching data for, such as fire zone, public works division, ward, tract, council district. By using drop(), I was able to “clean” out the dataset into the important sections that aligned with the research.

This left the neighborhood, the total count of guns, the types of guns, the date, and the zone

```
   _id                                        address  total_count  \
0    1     1700 BLOCK ARLINGTON AVE PITTSBURGH, PA 15210            2
1    2       BREVET WAY & MINTON ST PITTSBURGH, PA 15204            1
2    3    TERRACE ST & WHITRIDGE ST PITTSBURGH, PA 15213            1
3    4         3500 BLOCK GERBER AVE PITTSBURGH, PA 15212            1
4    5           7500 BLOCK KELLY ST PITTSBURGH, PA 15208            1
5    6            1900 BLOCK 5TH AVE PITTSBURGH, PA 15219            1
6    7       400 BLOCK S NEGLEY AVE PITTSBURGH, PA 15232            1
7    8       5200 BLOCK DRESDEN WAY PITTSBURGH, PA 15201            2
8    9          1100 BLOCK GRAND AVE PITTSBURGH, PA 15212            1
9   10  1600 BLOCK FALLOWFIELD AVE PITTSBURGH, PA 15216            1

   other_count  pistol_count  revolver_count  rifle_count  shotgun_count  \
0            0             1               1            0              0
1            0             1               0            0              0
2            0             1               0            0              0
3            0             1               0            0              0
4            0             1               0            0              0
5            0             1               0            0              0
6            0             1               0            0              0
7            0             2               0            0              0
8            0             1               0            0              0
9            0             1               0            0              0

   year  month  dow          neighborhood police_zone
0  2015      1    0  Mount Oliver Borough           3
1  2015      1    0              Sheraden           6
2  2015      1    0       Terrace Village           2
3  2015      1    0      Brighton Heights           1
4  2015      1    0                   NaN           5
5  2015      1    0              Westwood           6
6  2015      1    1          East Liberty           5
7  2015      1    1   Upper Lawrenceville           2
8  2015      1    1     Marshall–Shadeland           1
9  2015      1    1             Beechview           6
```

```
columns_to_drop = ['latitude', 'longitude', 'fire_zone', 'public_works_division', 'ward', 'tract', 'council_district']
df_cleaned = df.drop(columns=columns_to_drop, errors='ignore')

print(df_cleaned.head(10))
```

Firearm Seizures in Pittsburgh



A bar chart was then created to display the top 10 neighborhoods with the most firearms seized. It indicated that Homewood South had the most firearms seized.

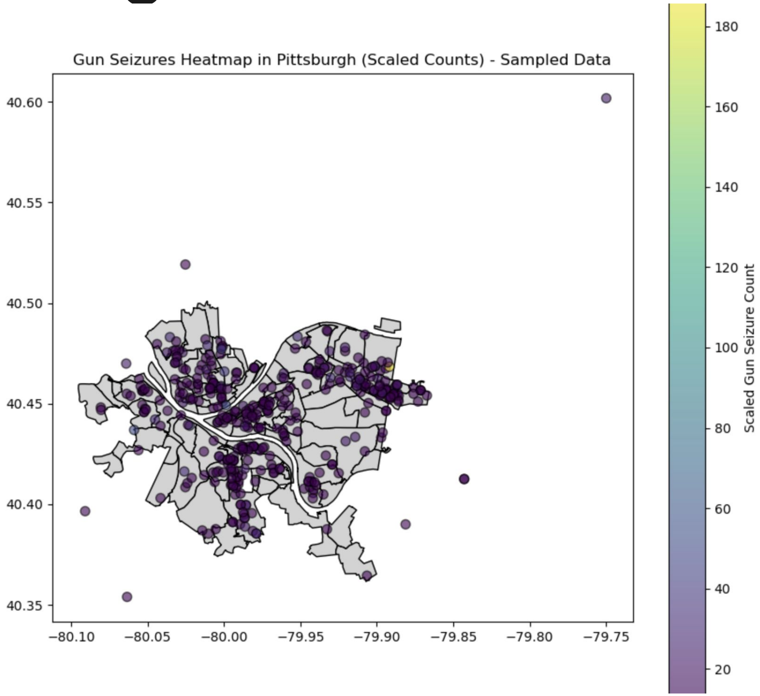
In this code, I used pandas and matplotlib to make the chart and implement the data into it

Firearm Seizures in Pittsburgh

I then created a heat map in order to create a new visualization for the data so that it would be easier to see exactly where the weapons were taken the most

For this geopandas, pandas, matplotlib, and numpy were imported

In conclusion through this and the bar chart, It shows that Homewood South and Southside Flats have the most seizures and are not good places to live in in my opinion. These areas are already known for not being the safest areas in general, and this data supports that.



Code for Bar Chart and Map

```python
import pandas as pd
import matplotlib.pyplot as plt

data_path = "firearms.csv"
df = pd.read_csv(data_path)

columns_to_drop = ['latitude', 'longitude', 'fire_zone', 'public_works_division', 'ward', 'tract', 'council_district']
df_cleaned = df.drop(columns=columns_to_drop, errors='ignore')

neighborhood_totals = df_cleaned.groupby('neighborhood')['total_count'].sum().sort_values(ascending=False)

top_neighborhoods = neighborhood_totals.head(10)  # Top 10 neighborhoods
top_neighborhoods.plot(kind='bar', figsize=(10, 6), color='skyblue', edgecolor='black')

plt.title('Top 10 Neighborhoods by Firearm Seizures')
plt.xlabel('Neighborhood')
plt.ylabel('Total Guns Seized')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()

plt.show()
```

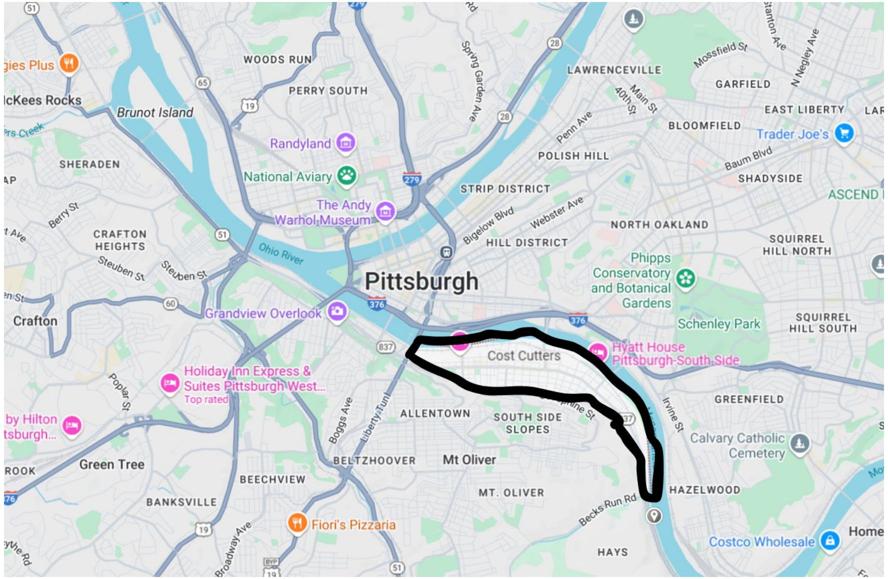
```python
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import Normalize

gdf = gpd.read_file('pittsburgh.geojson')

gun_seizures = pd.read_csv('firearms.csv')

gun_seizures_cleaned = gun_seizures[['latitude', 'longitude', 'total_count']].dropna()
gun_seizures_cleaned = gun_seizures_cleaned[~gun_seizures_cleaned.isin([np.inf, -np.inf]).any(axis=1)]
gun_seizures_cleaned = gun_seizures_cleaned[gun_seizures_cleaned['total_count'] > 0]

sampled_data = gun_seizures_cleaned.sample(frac=0.1, random_state=42)

sampled_data['scaled_count'] = sampled_data['total_count'] * 100

print("Scaled Count Range:", sampled_data['scaled_count'].min(), "-", sampled_data['scaled_count'].max())

fig, ax = plt.subplots(figsize=(10, 10))
gdf.plot(ax=ax, color='lightgray', edgecolor='black')

norm = Normalize(vmin=sampled_data['scaled_count'].min(), vmax=sampled_data['scaled_count'].max())

scatter = ax.scatter(sampled_data['longitude'], sampled_data['latitude'],
                     c=sampled_data['scaled_count'], cmap='viridis', s=50, alpha=0.6, edgecolors='k', norm=norm)

fig.colorbar(scatter, ax=ax, label='Scaled Gun Seizure Count')
plt.title('Gun Seizures Heatmap in Pittsburgh (Scaled Counts) - Sampled Data')
plt.show()
```

Conclusion

To declare which neighborhood is the worst overall we compared all of our top 5 contenders as well as which neighborhood(s) occurred the most out of our collective list.

We decided that **South Side Flats** is our choice for the worst neighborhood.

This is because it was the only neighborhood to fall within all of our top 3 neighborhoods.



Aidans working heat map slides

```
import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt
import requests
from io import StringIO
import matplotlib.colors as mcolors

# Load the CSV data into a pandas DataFrame
data = pd.read_csv('data.csv')

# Group by the 'incident_zip' column and count occurrences
```