DC Food & Nutrition Services Analysis with Images

This notebook includes data cleaning, geospatial preparation, interactive visualizations, and static image generation for reports.

Group: 2

```
pip install pandas plotly kaleido
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: plotly in /usr/local/lib/python3.11/dist-packages (5.24.1)
Collecting kaleido
  Downloading kaleido-0.2.1-py2.py3-none-manylinux1_x86_64.whl.metadata (15 kB)
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.11/dist-packages (from plotly) (9.1.2)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from plotly) (24.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas) (1.17.0)
Downloading kaleido-0.2.1-py2.py3-none-manylinux1_x86_64.whl (79.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.9/79.9 MB 13.1 MB/s eta 0:00:00
Installing collected packages: kaleido
Successfully installed kaleido-0.2.1
```

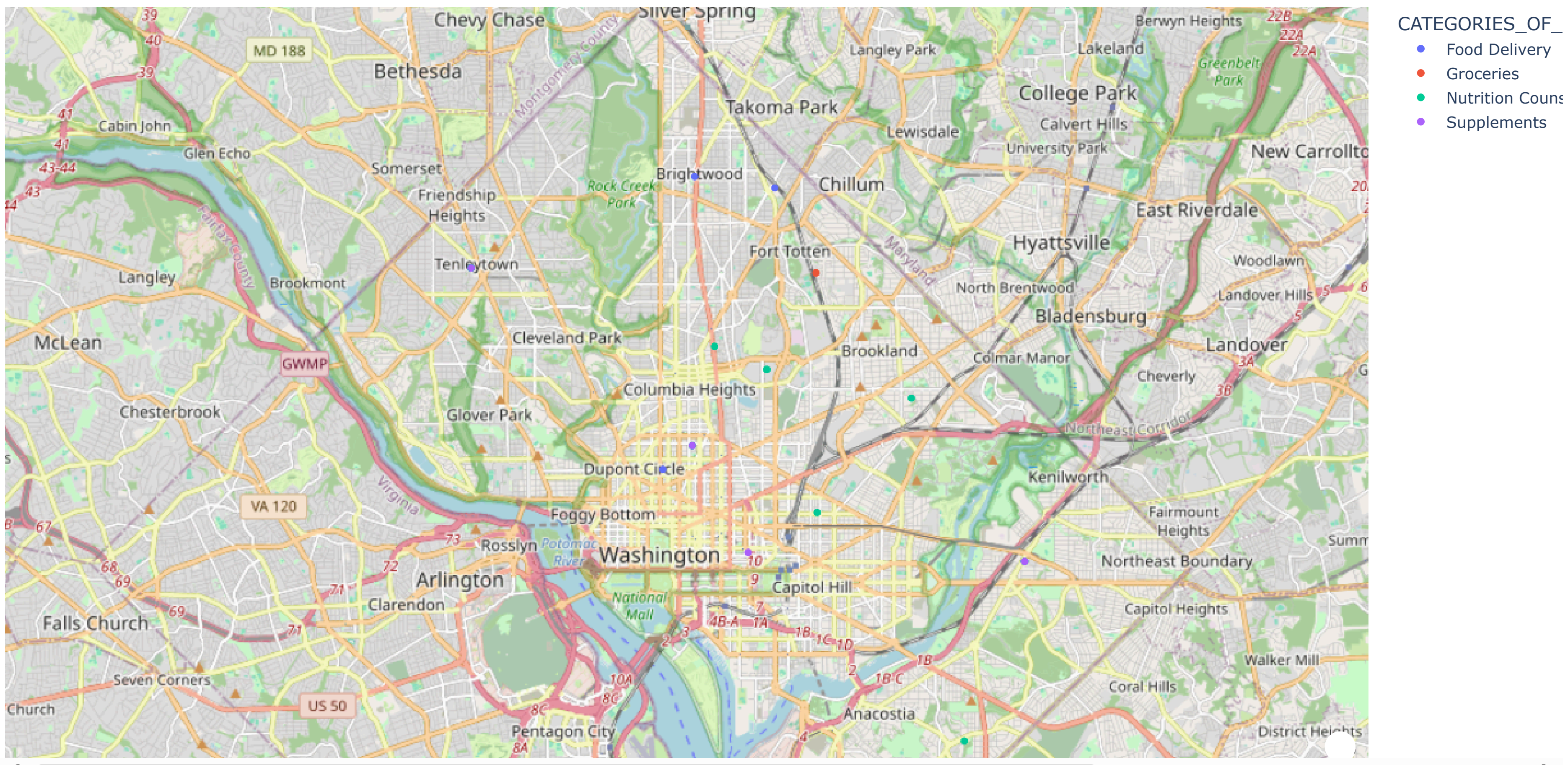
```
import pandas as pd
import plotly.express as px
import plotly.figure_factory as ff
import plotly.io as pio
```

Load and Clean Dataset

```
# Load data
df = pd.read_csv("Food_and_Nutrition.csv")

# Drop fully null columns
df.dropna(axis=1, how='all', inplace=True)

# Fill missing email values
df['EMAIL'] = df['EMAIL'].fillna('Not Provided')

# Rename and ensure proper data types
df.rename(columns={
    'X': 'Longitude_Projected',
    'Y': 'Latitude_Projected',
    'MAR_LATITUDE': 'Latitude',
    'MAR_LONGITUDE': 'Longitude'
}, inplace=True)

df['Latitude'] = pd.to_numeric(df['Latitude'], errors='coerce')
df['Longitude'] = pd.to_numeric(df['Longitude'], errors='coerce')
df.dropna(subset=['Latitude', 'Longitude'], inplace=True)

df.reset_index(drop=True, inplace=True)
```

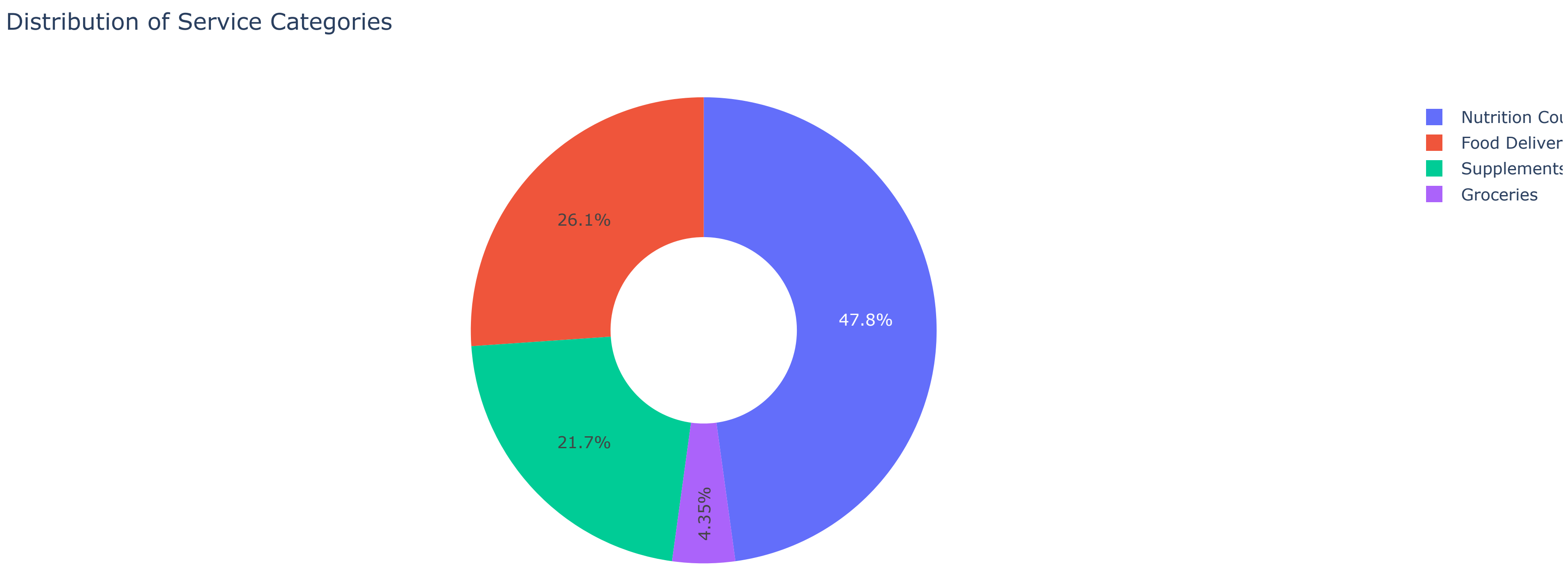
Interactive Map Visualization

```
df.reset_index(drop=True, inplace=True)
fig = px.scatter_mapbox(
    df, lat="Latitude", lon="Longitude",
    hover_name="ORGANIZATION",
    hover_data=["CATEGORIES_OF_SERVICE", "FULL_ADDRESS", "PHONE"],
    color="CATEGORIES_OF_SERVICE",
    zoom=11, height=600
)
fig.update_layout(mapbox_style="open-street-map", margin={"r":0,"t":0,"l":0,"b":0})
fig.write_html("food_nutrition_map.html")
fig.show()
```



Category Distribution

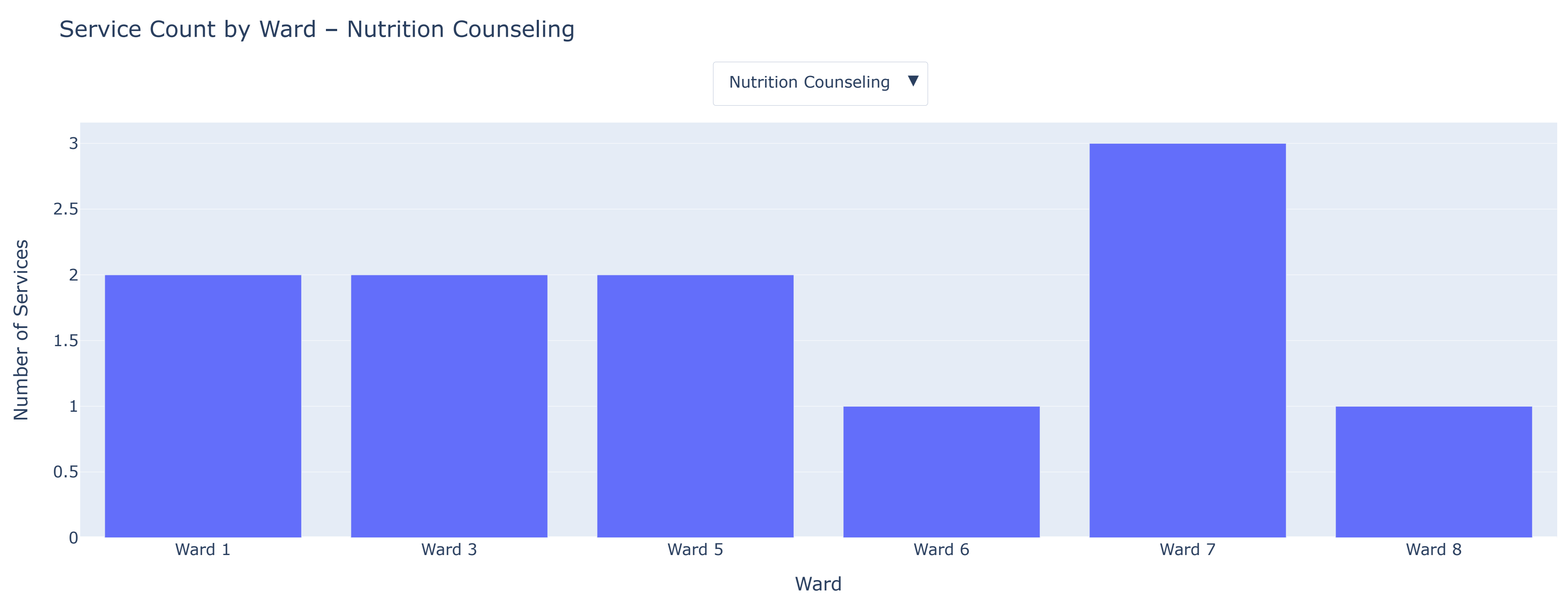
```
import plotly.express as px

# Prepare the data
category_counts = df['CATEGORIES_OF_SERVICE'].value_counts().reset_index()
category_counts.columns = ['Service Category', 'Count']  # Rename columns

# Create the pie chart
fig_pie = px.pie(
    category_counts,
    names='Service Category',
    values='Count',
    title='Distribution of Service Categories',
    hole=0.4  # Optional: donut-style chart
)

# Save and show the figure
fig_pie.write_html("service_category_distribution_pie.html")
fig_pie.show()
```



Services by Ward

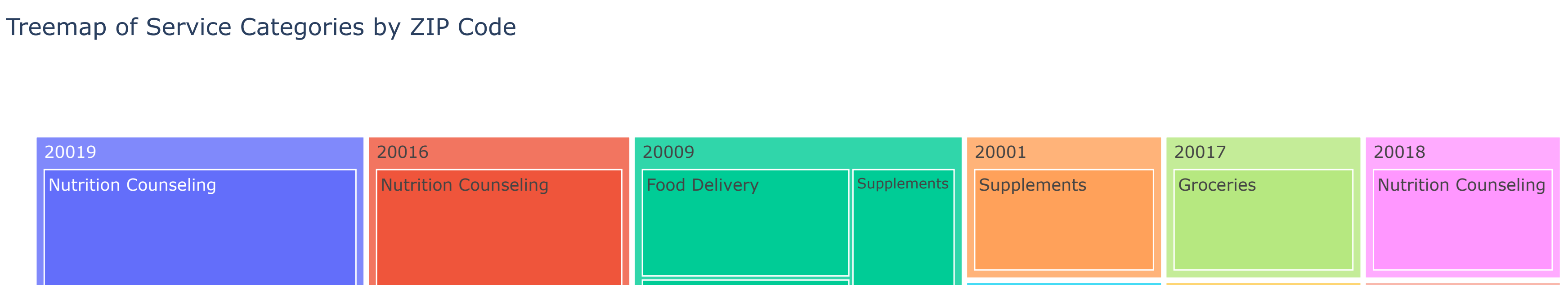
```
import plotly.graph_objects as go
import plotly.io as pio

df.dropna(axis=1, how='all', inplace=True)
df['EMAIL'] = df['EMAIL'].fillna('Not Provided')

df.rename(columns={
    'X': 'Longitude_Projected',
    'Y': 'Latitude_Projected',
    'MAR_LATITUDE': 'Latitude',
    'MAR_LONGITUDE': 'Longitude'
}, inplace=True)

df['Latitude'] = pd.to_numeric(df['Latitude'], errors='coerce')
df['Longitude'] = pd.to_numeric(df['Longitude'], errors='coerce')
df.dropna(subset=['Latitude', 'Longitude'], inplace=True)
df.reset_index(drop=True, inplace=True)
```

```
df['MAR_WARD'] = df['MAR_WARD'].astype(str)
df['CATEGORIES_OF_SERVICE'] = df['CATEGORIES_OF_SERVICE'].astype(str)

grouped = df.groupby(['MAR_WARD', 'CATEGORIES_OF_SERVICE']).size().reset_index(name='Count')
grouped_all = df.groupby(['MAR_WARD']).size().reset_index(name='Count')
grouped_all['CATEGORIES_OF_SERVICE'] = 'All Services'
grouped_combined = pd.concat([grouped, grouped_all], ignore_index=True)
categories = grouped_combined['CATEGORIES_OF_SERVICE'].unique()
fig = go.Figure()
initial_data = grouped_combined[grouped_combined['CATEGORIES_OF_SERVICE'] == 'All Services']
fig.add_trace(go.Bar(
    x=initial_data['MAR_WARD'],
    y=initial_data['Count'],
    name='All Services'
))
dropdown_buttons = []
for category in categories:
    filtered = grouped_combined[grouped_combined['CATEGORIES_OF_SERVICE'] == category]
    dropdown_buttons.append(dict(
        label=category,
        method="update",
        args=[{
            "x": [filtered['MAR_WARD']],
            "y": [filtered['Count']],
            "type": "bar"
        }, {
            "title": f"Service Count by Ward – {category}"
        }]
    ))
fig.update_layout(
    title="Service Count by Ward – All Services",
    updatemenus=[{
        'buttons': dropdown_buttons,
        'direction': 'down',
        'showactive': True,
        'x': 0.5,
        'y': 1.15,
        'xanchor': 'center',
        'yanchor': 'top'
    }],
    xaxis_title="Ward",
    yaxis_title="Number of Services",
    margin=dict(t=130, b=80)
)
fig.write_html("ward_service_dropdown.html")
fig.show()
```

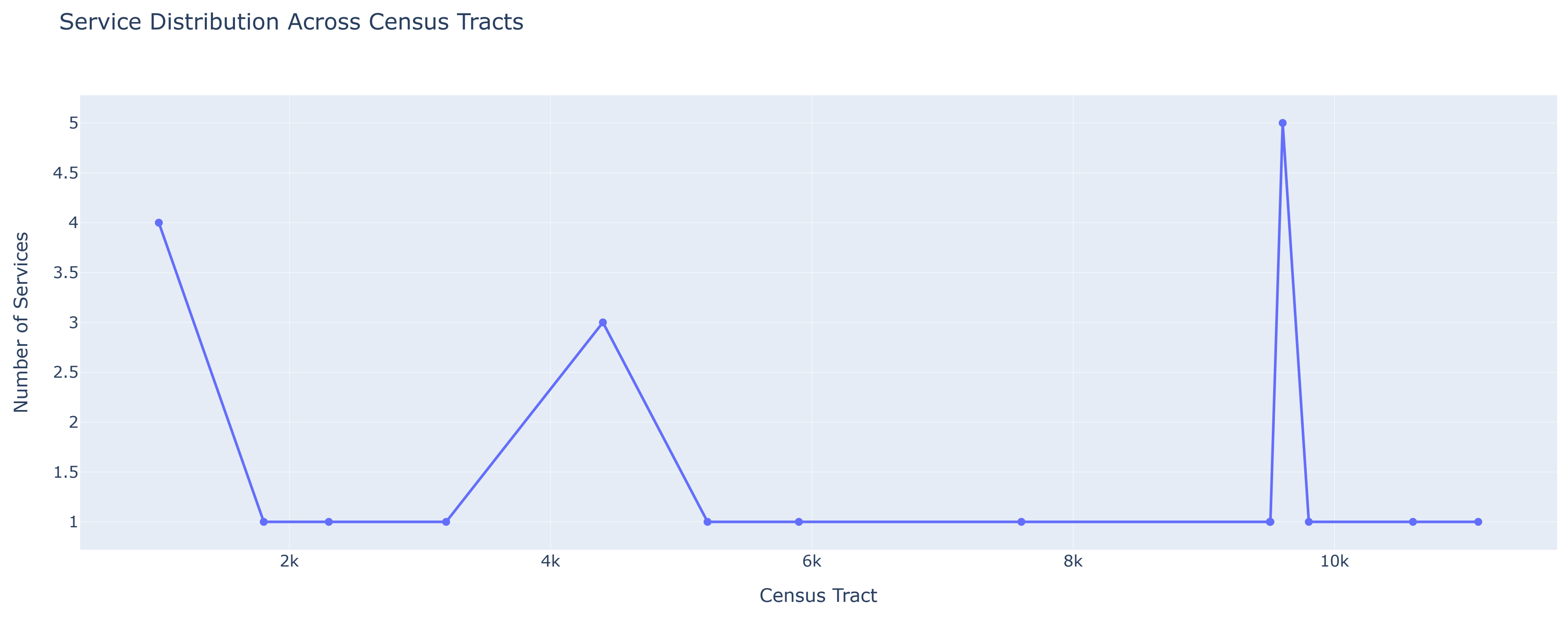


Geospatial Density Map

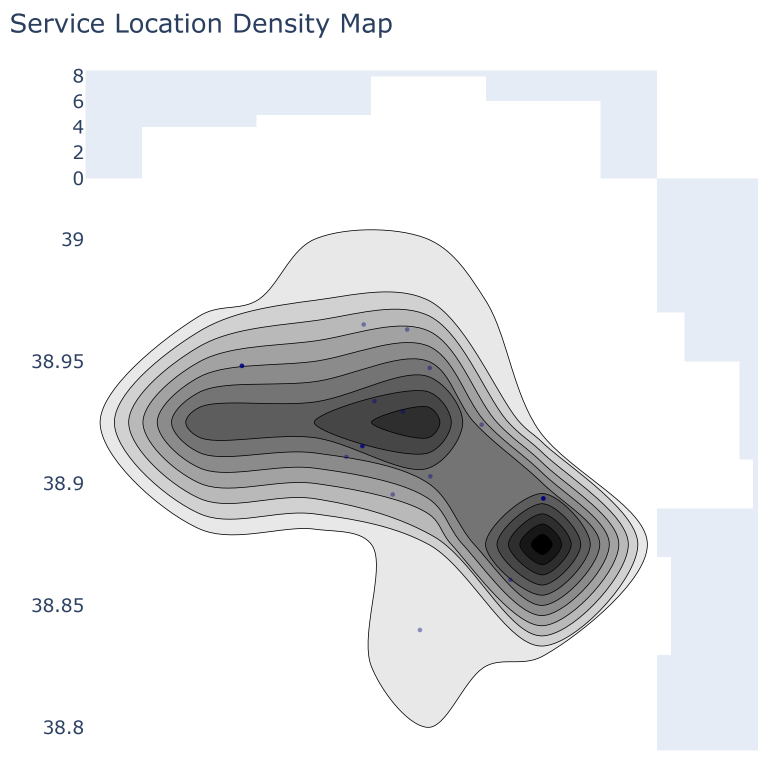
```
# Treemap: Service Categories by ZIP Code
fig_treemap = px.treemap(
    df,
    path=['ZIPCODE', 'CATEGORIES_OF_SERVICE'],
    title='Treemap of Service Categories by ZIP Code',
)
fig_treemap.write_html("treemap_service_by_zip.html")
fig_treemap.show()
```



```
import plotly.express as px

# Prepare data: count services by MAR_CENSUS_TRACT
tract_counts = df['MAR_CENSUS_TRACT'].value_counts().sort_index().reset_index()
tract_counts.columns = ['Census Tract', 'Service Count']

# Create a line plot to simulate a trend across geographic tracts
fig_tract = px.line(
    tract_counts,
    x='Census Tract',
    y='Service Count',
    title='Service Distribution Across Census Tracts',
    markers=True,
    labels={'Census Tract': 'Census Tract', 'Service Count': 'Number of Services'}
)

fig_tract.write_html("trend_service_by_tract.html")
fig_tract.show()
```



```
fig_density = ff.create_2d_density(
    x=df['Longitude'], y=df['Latitude'],
    colorscale='Hot', hist_color='rgba(255,255,255,0)', point_size=3
)
fig_density.update_layout(title="Service Location Density Map", height=600)
fig_density.write_html("geospatial_density_analysis.html")
fig_density.show()
fig_density.write_html("density_map.html")

#pio.write_image(fig_density, "density_map.png")
```


Service Location Density Map