```
import pandas as pd

df = pd.read_csv('C:\\Users\\chuck\\Documents\\LVFD\\
Fire_Department_Incident_Count_raw.csv')

#Display the firsat few rows of the dataframe
df.head()
```

```
             Response_Date  Year_Extracted  Month_Extracted Event_Type
\
0  2012/01/01 05:30:00+00            2012                1    Medical

1  2012/01/01 05:31:00+00            2012                1    Medical

2  2012/01/01 05:32:00+00            2012                1    Medical

3  2012/01/01 14:57:00+00            2012                1    Medical

4  2012/01/01 14:58:00+00            2012                1    Medical


  Division     Station Radio_Name                        Location  \
0   Div 01   Station 1       R201        100 block of FREMONT ST
1   Div 01   Station 3         R3        900 block of Pyramid Dr
2   Div 10   Station 2         R2          9000 block of Alta Dr
3   Div 01  Station 10        R10  Cordova St  &  E St Louis Ave
4   Div 10  Station 44        E44      6200 block of Clarice Ave

          Location_Coordinates
0   (36.17168427, -115.146347)
1  (36.18119431, -115.1897278)
2  (36.16799927, -115.2944489)
3  (36.14778519, -115.1427994)
4  (36.17484665, -115.2262726)
```

```
import matplotlib.pyplot as plt

#Convert Response_Date to DateTime format
df["Response_Date"] = pd.to_datetime(df["Response_Date"])

#Group by "Year_Extracted" and "Event_Type" and count the occurances
event_counts = df.groupby(["Year_Extracted",
"Event_Type"]).size().reset_index(name="Counts")

#Display the result
print(event_counts)
```

```
    Year_Extracted   Event_Type  Counts
0             2012         Fire    2976
1             2012      Medical   79827
2             2012        Other     989
3             2012  Public Need    4222
```

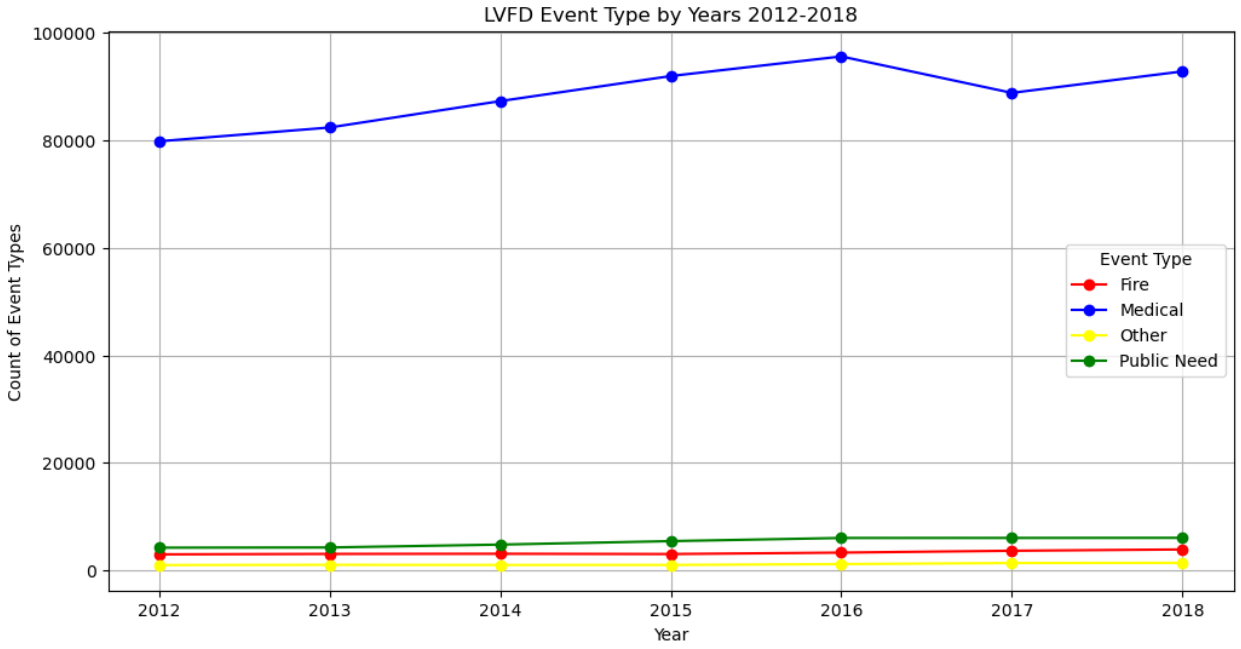
```
4              2013         Fire    3058
5              2013      Medical   82392
6              2013        Other    1026
7              2013  Public Need    4260
8              2014         Fire    3087
9              2014      Medical   87301
10             2014        Other    1001
11             2014  Public Need    4796
12             2015         Fire    3036
13             2015      Medical   91963
14             2015        Other    1006
15             2015  Public Need    5440
16             2016         Fire    3302
17             2016      Medical   95613
18             2016        Other    1162
19             2016  Public Need    6027
20             2017         Fire    3646
21             2017      Medical   88832
22             2017        Other    1365
23             2017  Public Need    6034
24             2018         Fire    3891
25             2018      Medical   92824
26             2018        Other    1396
27             2018  Public Need    6052
```

```
#Pivot the data to have years as rows and event types as columns
event_counts_pivot = event_counts.pivot(index= "Year_Extracted",
columns="Event_Type", values="Counts").fillna(0)

#Display the result
print(event_counts_pivot)
```

```
Event_Type      Fire  Medical  Other  Public Need
Year_Extracted
2012            2976    79827    989         4222
2013            3058    82392   1026         4260
2014            3087    87301   1001         4796
2015            3036    91963   1006         5440
2016            3302    95613   1162         6027
2017            3646    88832   1365         6034
2018            3891    92824   1396         6052
```

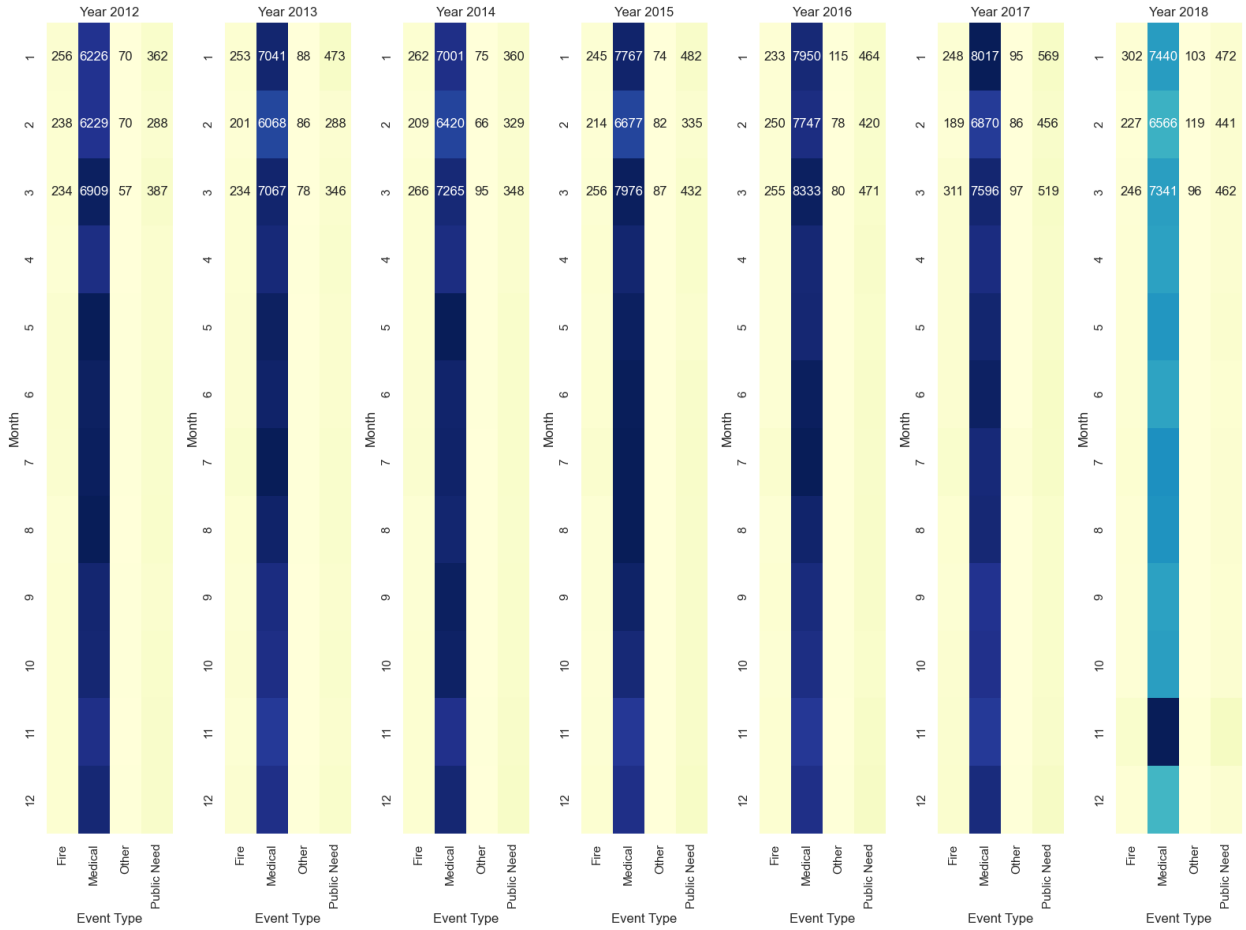
```
#Define color scheme
color_scheme = {
    "Fire": "red",
    "Medical": "blue",
    "Public Need": "green",
    "Other": "yellow"
}
```

```
#plot a multiple time series chart
plt.figure(figsize=(12, 6))

for column in event_counts_pivot.columns:
    plt.plot(event_counts_pivot.index, event_counts_pivot[column],
marker = "o", color =color_scheme[column], label=column)

plt.title("LVFD Event Type by Years 2012-2018")
plt.xlabel("Year")
plt.ylabel("Count of Event Types")
plt.legend(title = "Event Type")
plt.grid(True)
plt.show()
```



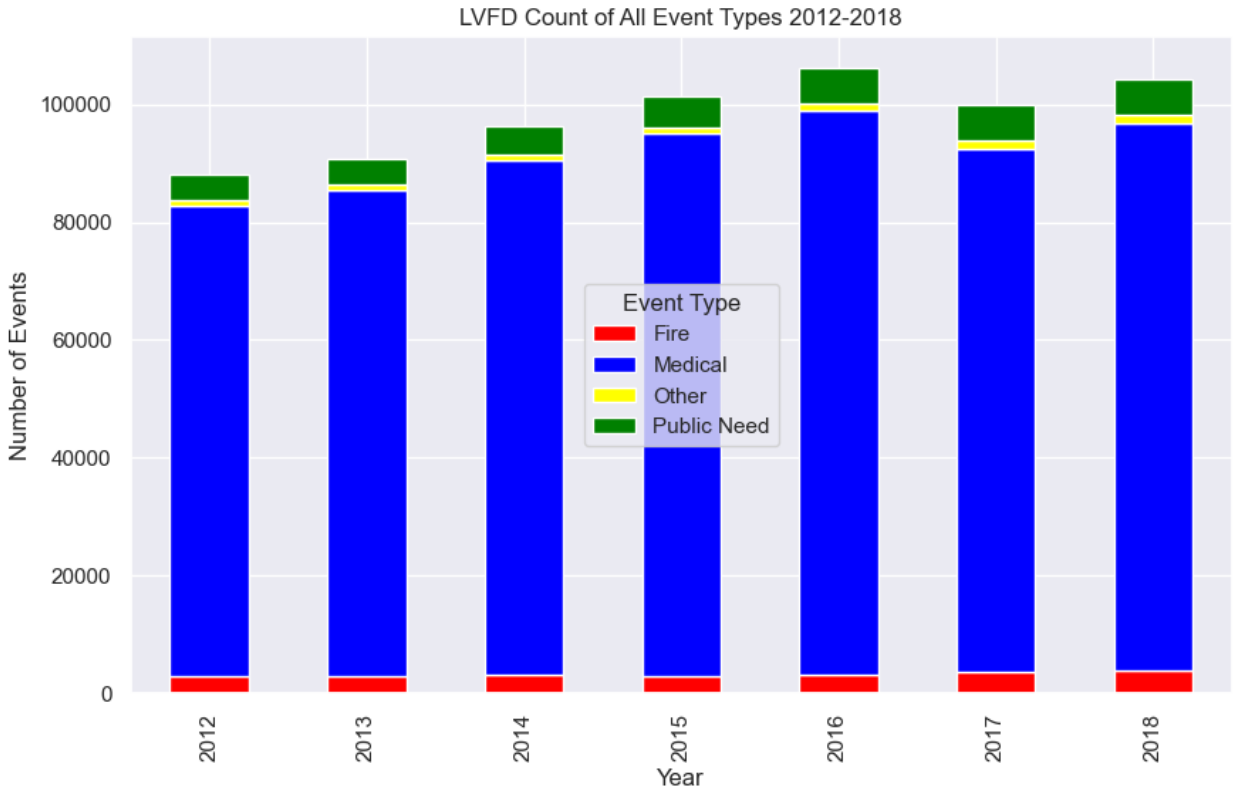

```
import seaborn as sns

# Pivot the DataFrame
event_counts_pivot = df.pivot_table(index='Month_Extracted',
columns=['Year_Extracted', 'Event_Type'], aggfunc='size',
fill_value=0)

# Plot the heatmap
plt.figure(figsize=(16, 12))
sns.set(font_scale=1)  # Adjust font size if needed

for year in
```

```
event_counts_pivot.columns.get_level_values('Year_Extracted').unique()
:
    plt.subplot(1,
len(event_counts_pivot.columns.get_level_values('Year_Extracted').uniq
ue()),
list(event_counts_pivot.columns.get_level_values('Year_Extracted').uni
que()).index(year) + 1)
    sns.heatmap(event_counts_pivot[year], cmap="YlGnBu", annot=True,
fmt="d", cbar=False)
    plt.title(f'Year {year}')
    plt.xlabel('Event Type')
    plt.ylabel('Month')

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



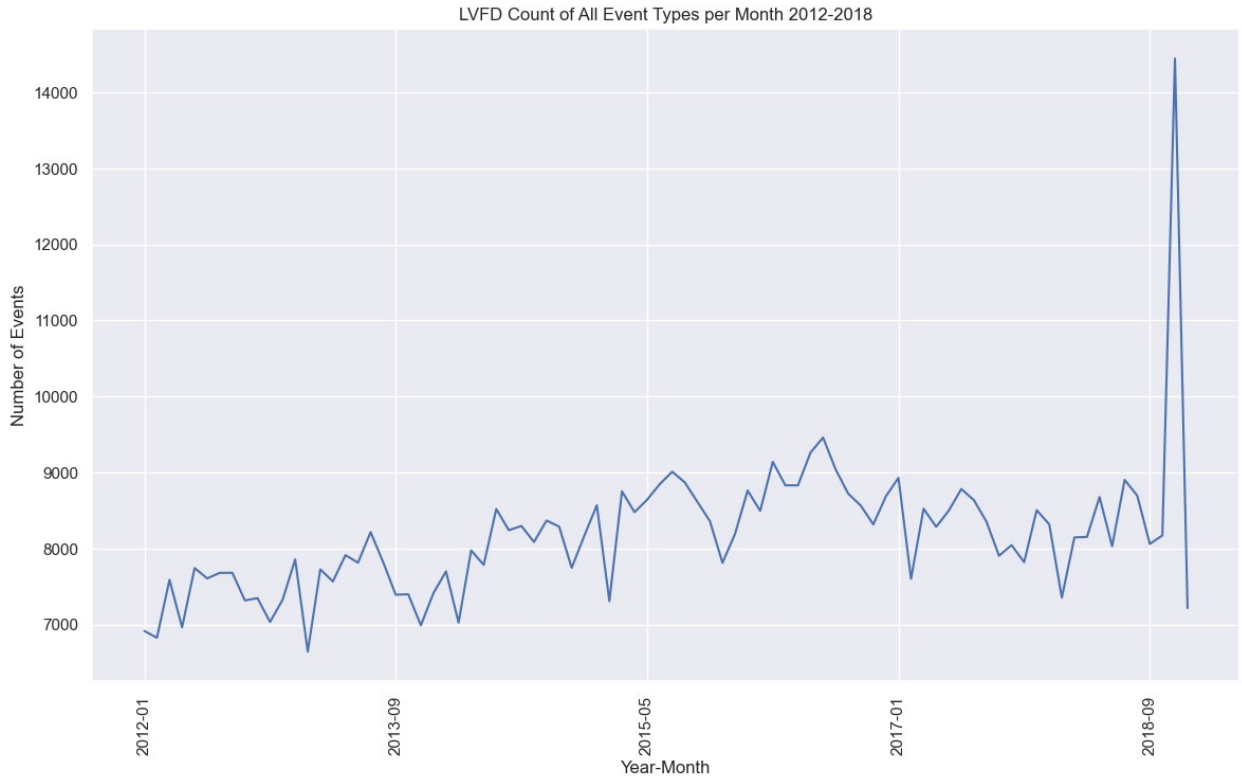

```
# Define the color scheme
color_scheme = {
    "Fire": "red",
```

```
    "Medical": "blue",
    "Public Need": "green",
    "Other": "yellow"
}

# Group by Year and Event_Type, then count the number of events
events_by_type_year = df.groupby(['Year_Extracted',
'Event_Type']).size().unstack().fillna(0)

# Map colors to event types
colors = [color_scheme.get(event_type, 'black') for event_type in
events_by_type_year.columns]

# Plot the data
events_by_type_year.plot(kind='bar', stacked=True, figsize=(10, 6),
color=colors)
plt.xlabel('Year')
plt.ylabel('Number of Events')
plt.title('LVFD Count of All Event Types 2012-2018')
plt.legend(title='Event Type')
plt.show()
```



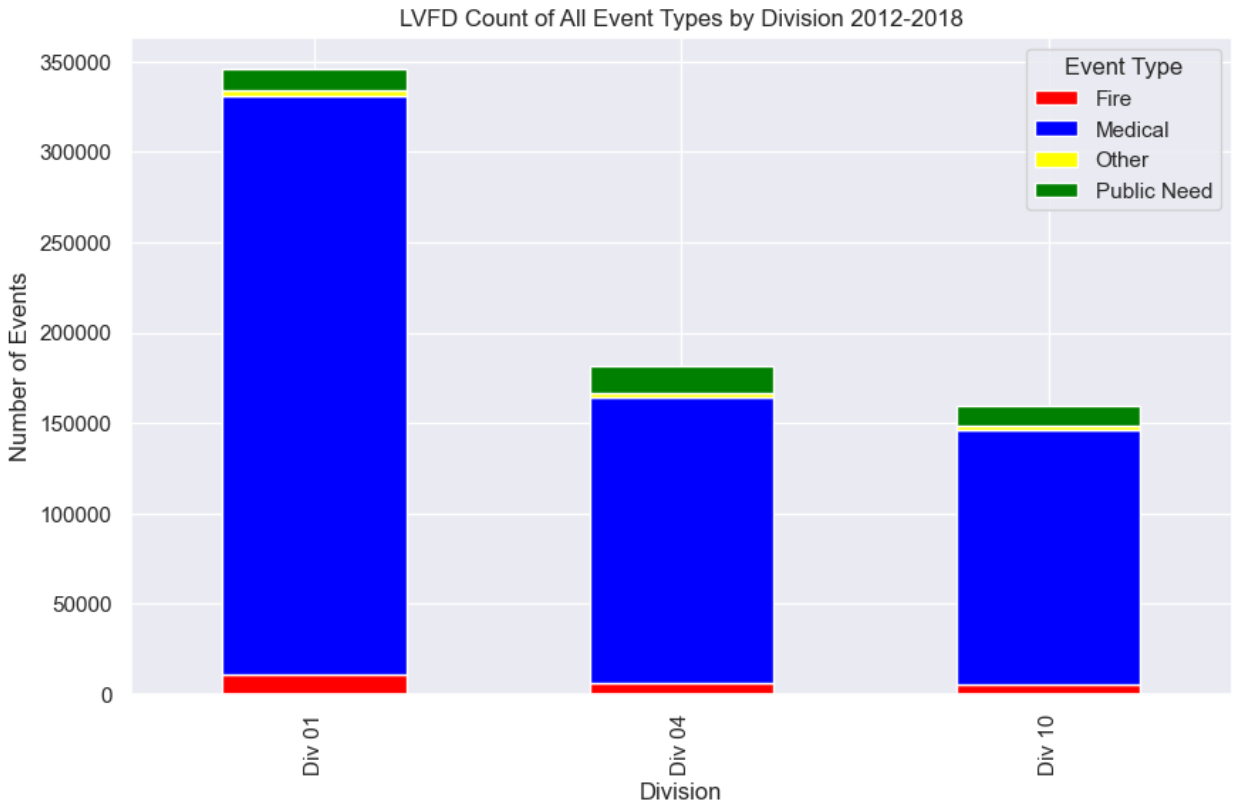

```
# Create a new column for combined Year and Month
df['Year_Month'] = df['Year_Extracted'].astype(str) + '-' +
df['Month_Extracted'].astype(str).str.zfill(2)
```

```
# Group by Year_Month and count the number of events
events_per_month_year = df['Year_Month'].value_counts().sort_index()

# Plot the data
plt.figure(figsize=(14, 8))
events_per_month_year.plot(kind='line')
plt.xlabel('Year-Month')
plt.ylabel('Number of Events')
plt.title('LVFD Count of All Event Types per Month 2012-2018')
plt.xticks(rotation=90)
plt.show()
```

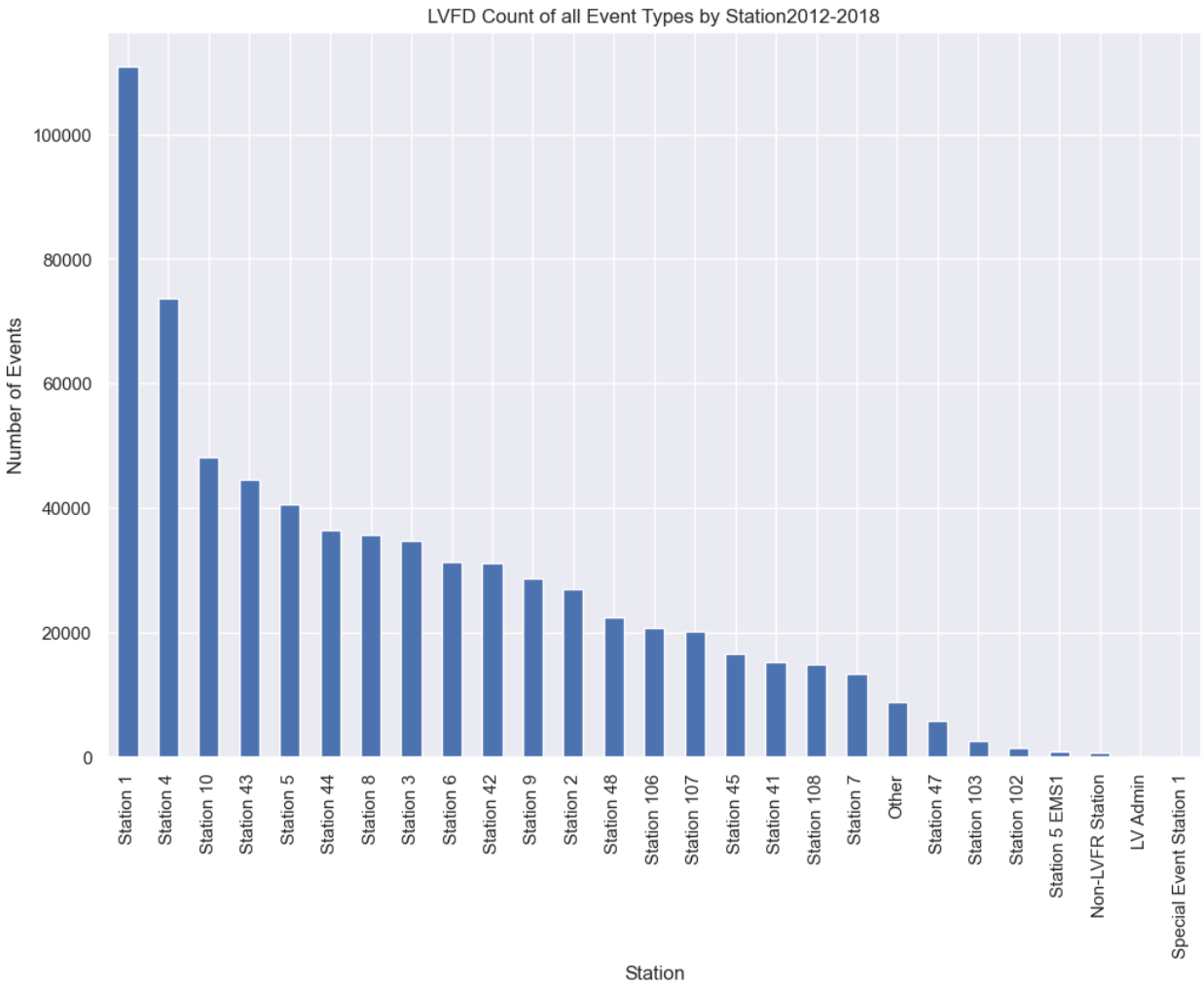



```
# Define the color scheme
color_scheme = {
    "Fire": "red",
    "Medical": "blue",
    "Public Need": "green",
    "Other": "yellow"
}

# Group by Division and Event_Type, then count the number of events
events_by_division_type = df.groupby(['Division',
'Event_Type']).size().unstack().fillna(0)
```

```
# Map colors to event types
colors = [color_scheme.get(event_type, 'black') for event_type in
events_by_division_type.columns]

# Plot the data
events_by_division_type.plot(kind='bar', stacked=True, figsize=(10,
6), color=colors)
plt.xlabel('Division')
plt.ylabel('Number of Events')
plt.title('LVFD Count of All Event Types by Division 2012-2018')
plt.legend(title='Event Type')
plt.show()
```

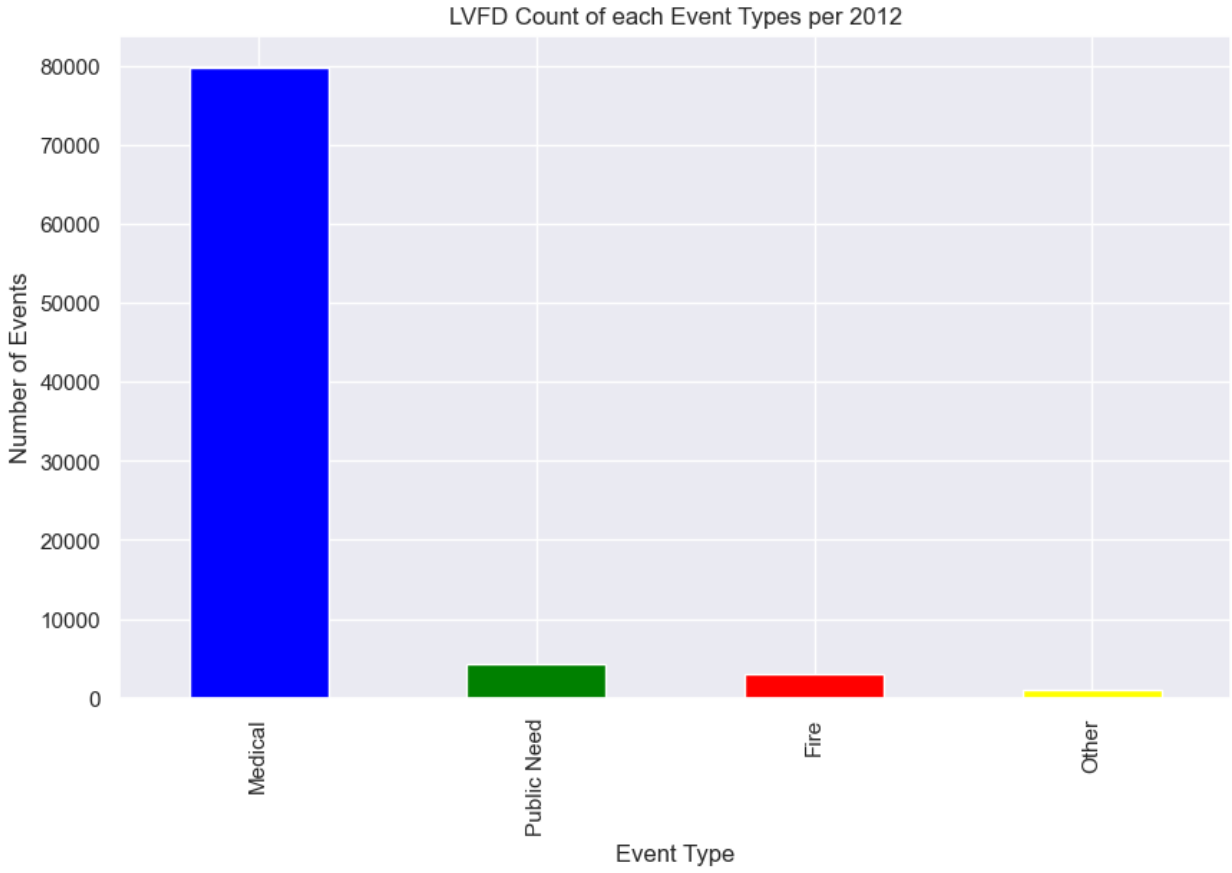


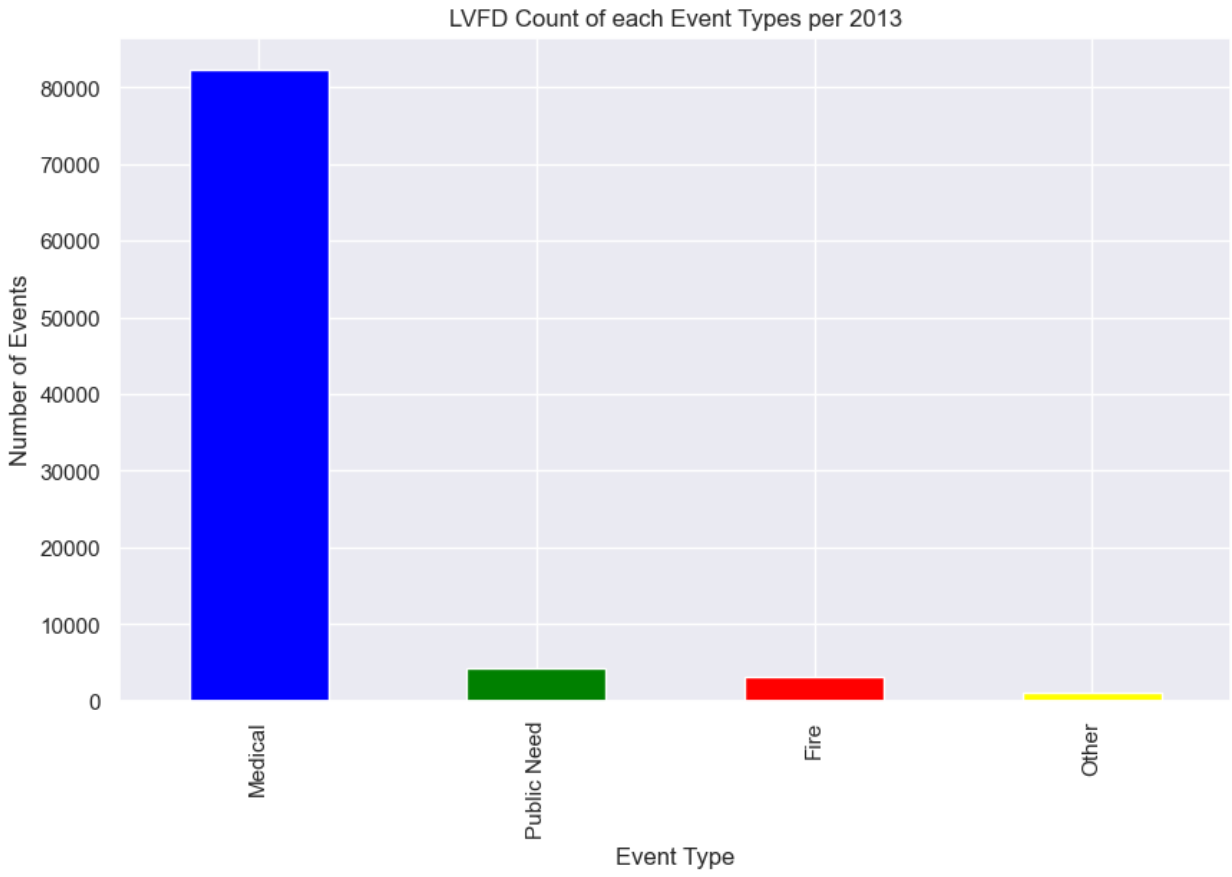

```
# Group by Station and count the number of events
events_by_station = df['Station'].value_counts()

# Plot the data
plt.figure(figsize=(12, 8))
events_by_station.plot(kind='bar')
plt.xlabel('Station')
plt.ylabel('Number of Events')
plt.title('LVFD Count of all Event Types by Station2012-2018')
plt.show()
```

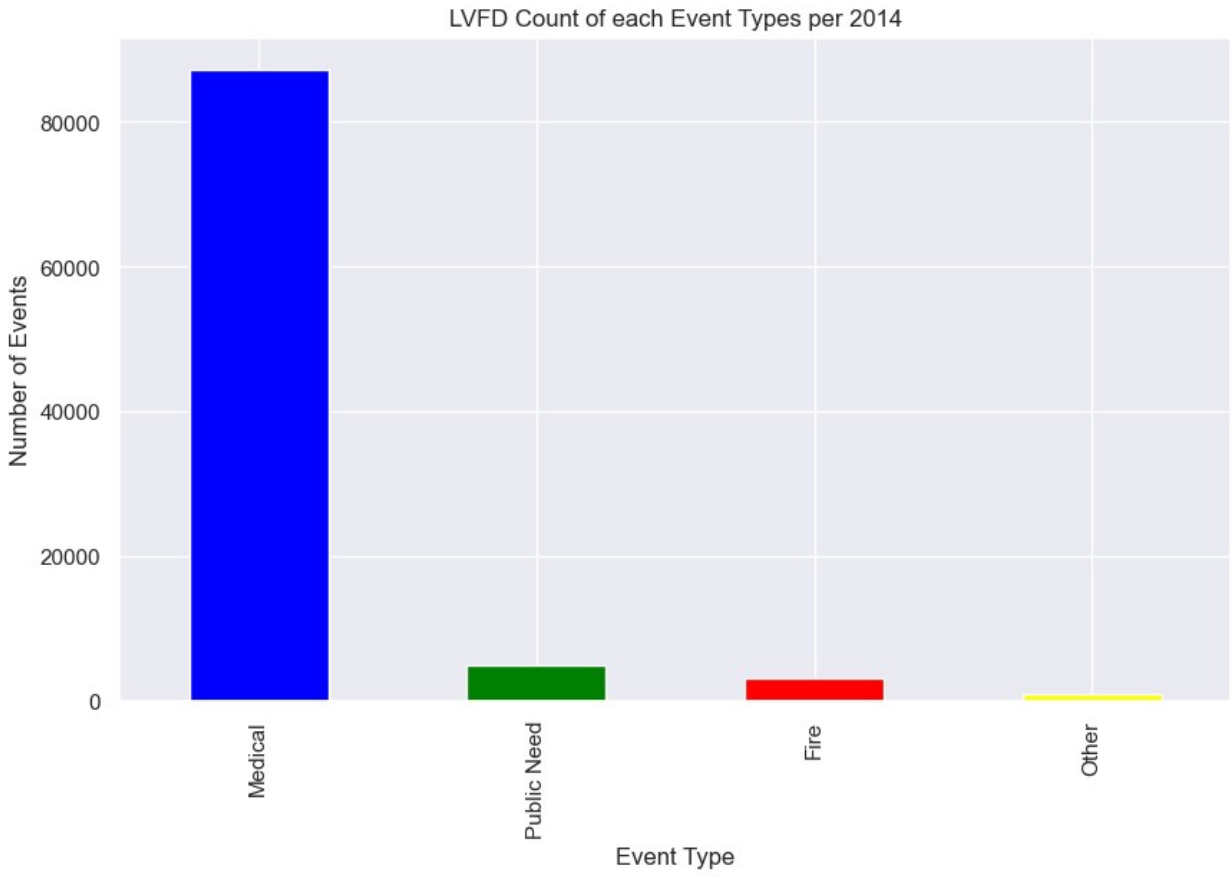


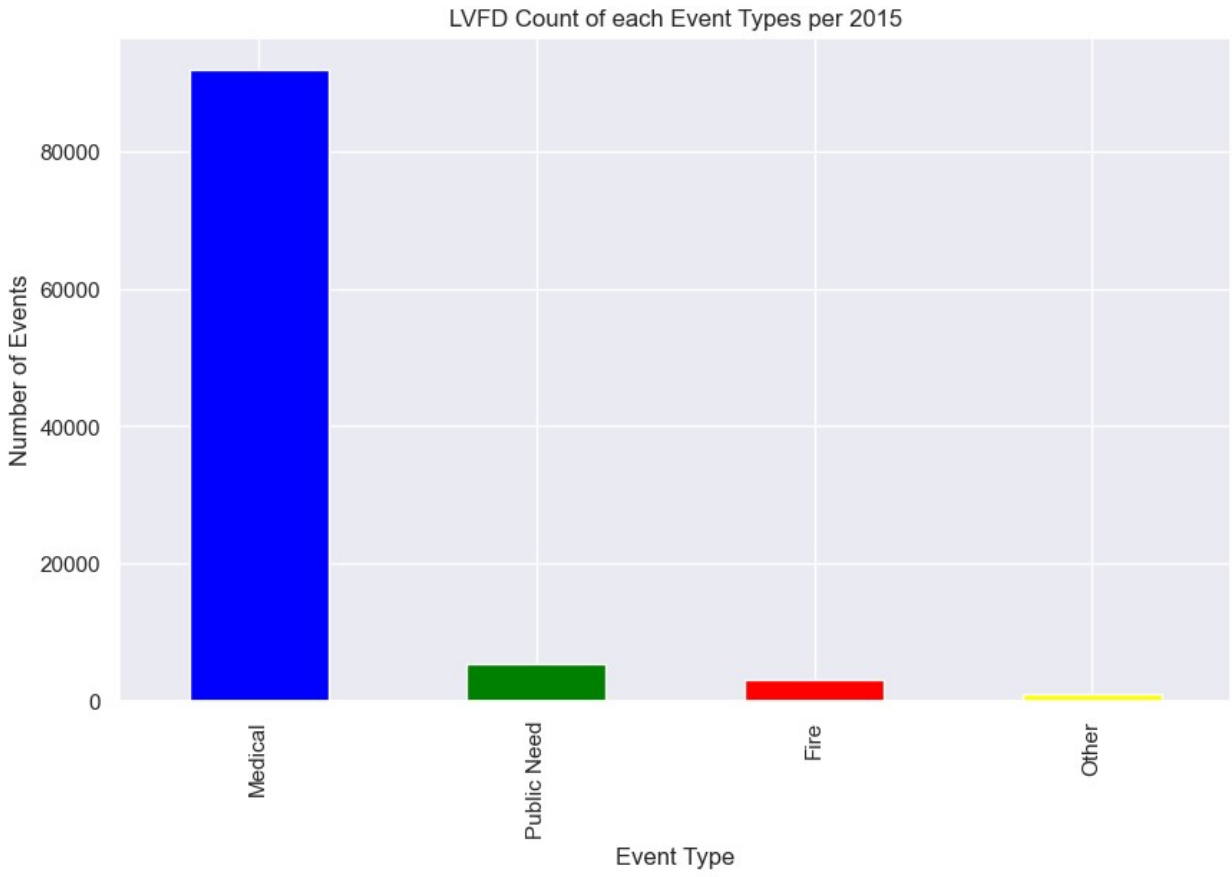

```
# Define the color scheme
color_scheme = {
    "Fire": "red",
    "Medical": "blue",
    "Public Need": "green",
    "Other": "yellow"
}

# Get a list of unique years
years = df['Year_Extracted'].unique()

# Create a plot for each year
for year in years:
    events_by_type_year = df[df['Year_Extracted'] == year]
['Event_Type'].value_counts()

    # Plot the data
    plt.figure(figsize=(10, 6))
    events_by_type_year.plot(kind='bar',
```

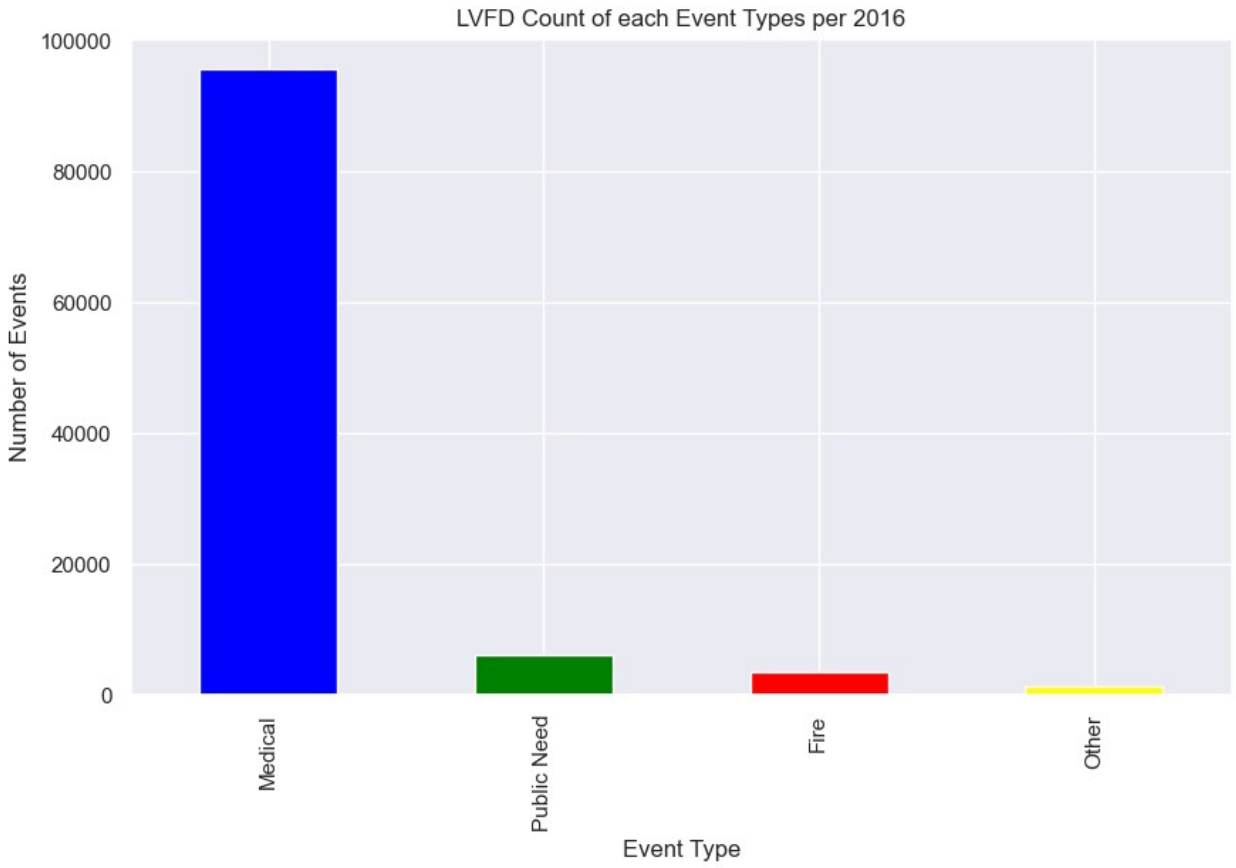
```
color=[color_scheme.get(event_type, 'black') for event_type in
events_by_type_year.index])
    plt.xlabel('Event Type')
    plt.ylabel('Number of Events')
    plt.title(f'LVFD Count of each Event Types per {year}')
    plt.show()
```

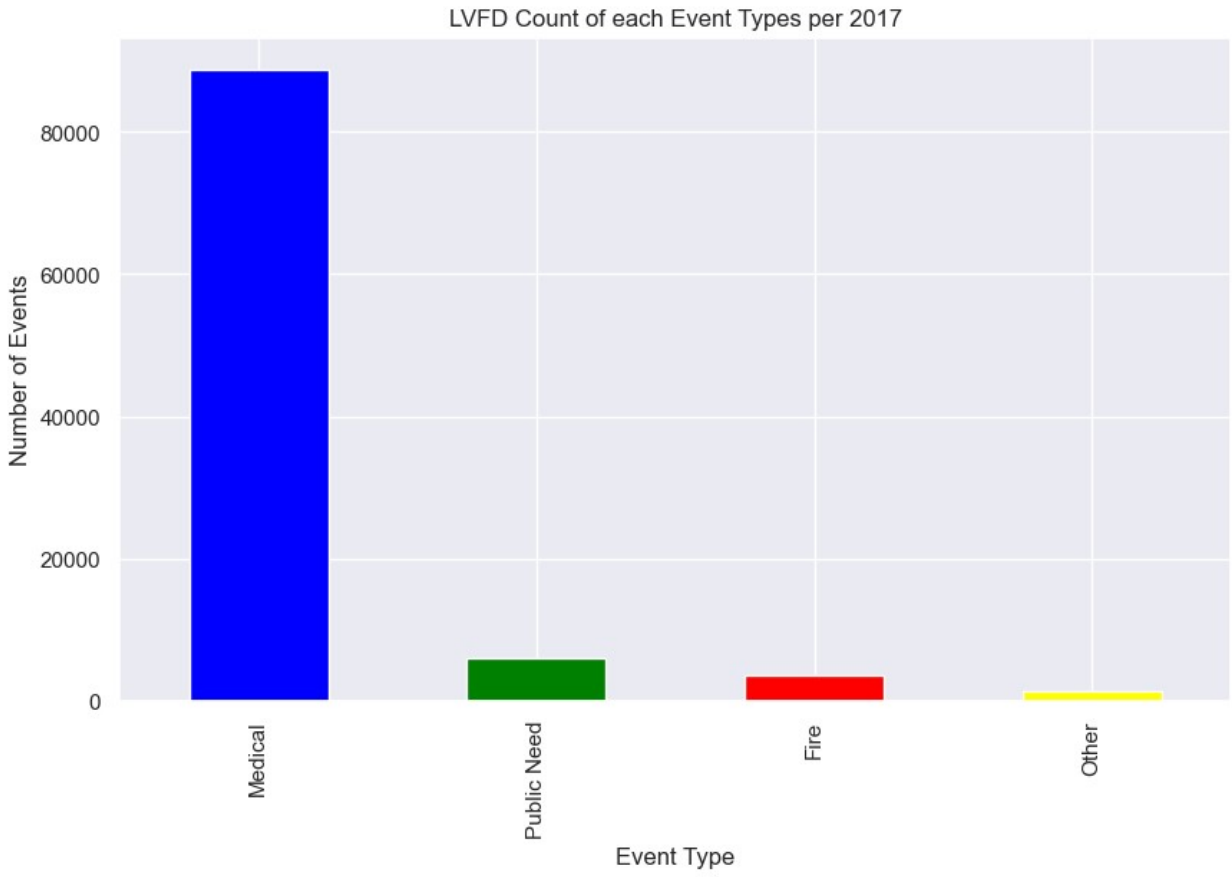


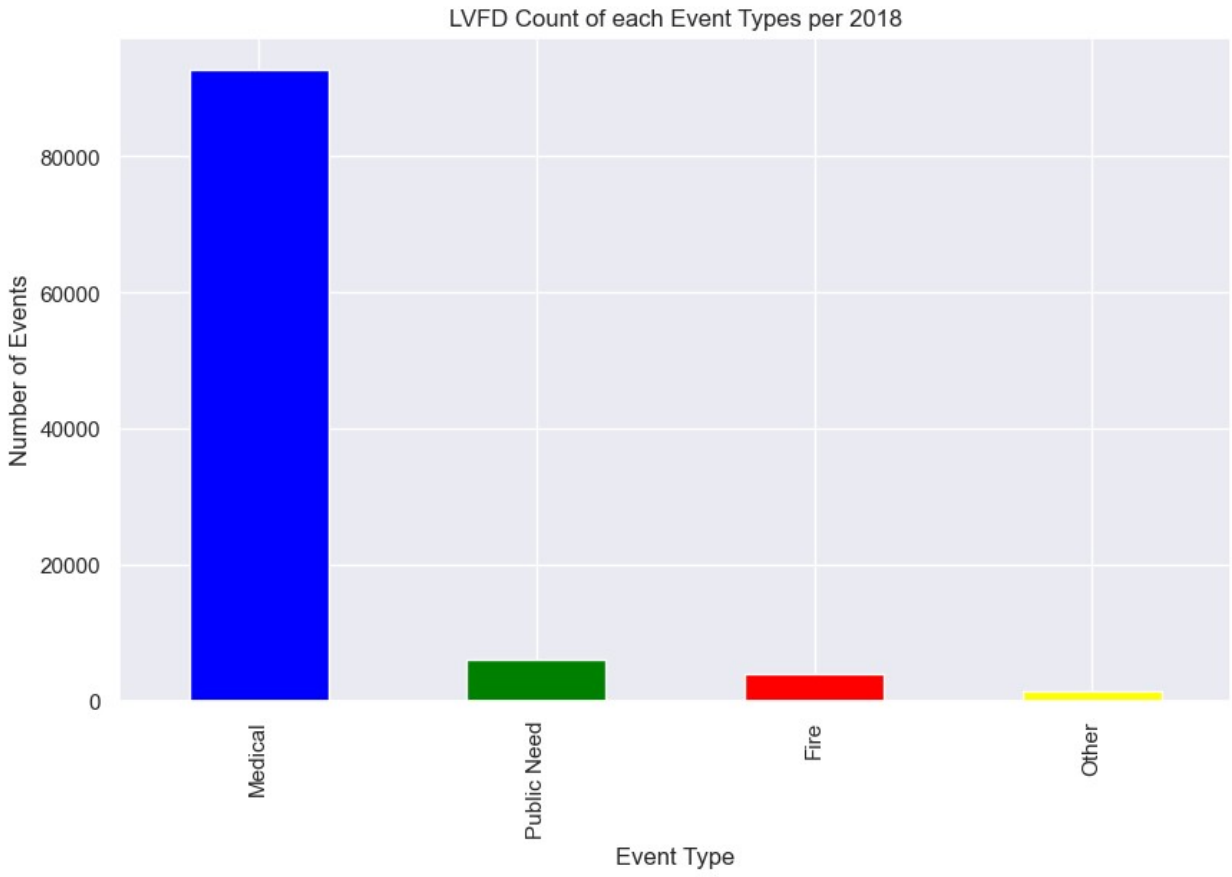

LVFD Count of each Event Types per 2013
Number of Events
80000
70000
60000
50000
40000
30000
20000
10000
0
Medical
Public Need
Fire
Other
Event Type


LVFD Count of each Event Types per 2014
Number of Events
80000
60000
40000
20000
0
Medical
Public Need
Fire
Other
Event Type

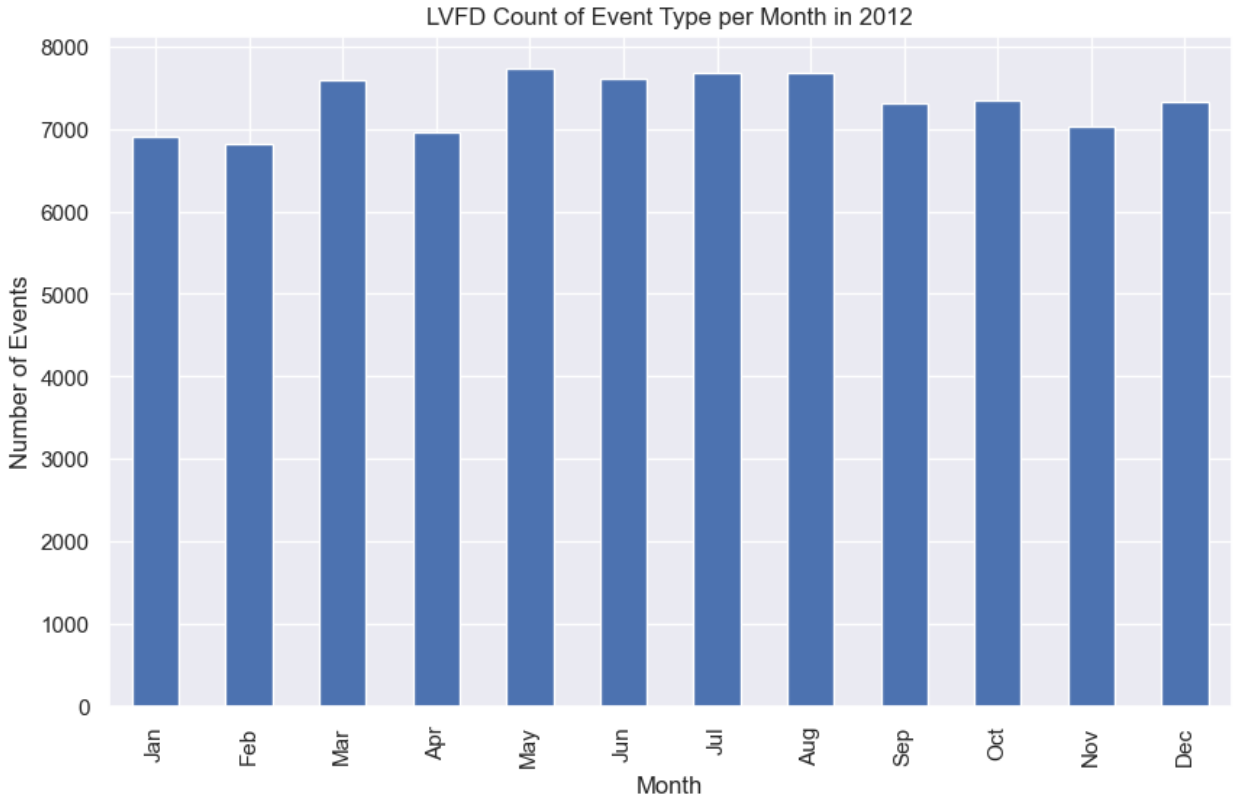

LVFD Count of each Event Types per 2015
Number of Events
80000
60000
40000
20000
0
Medical
Public Need
Fire
Other
Event Type

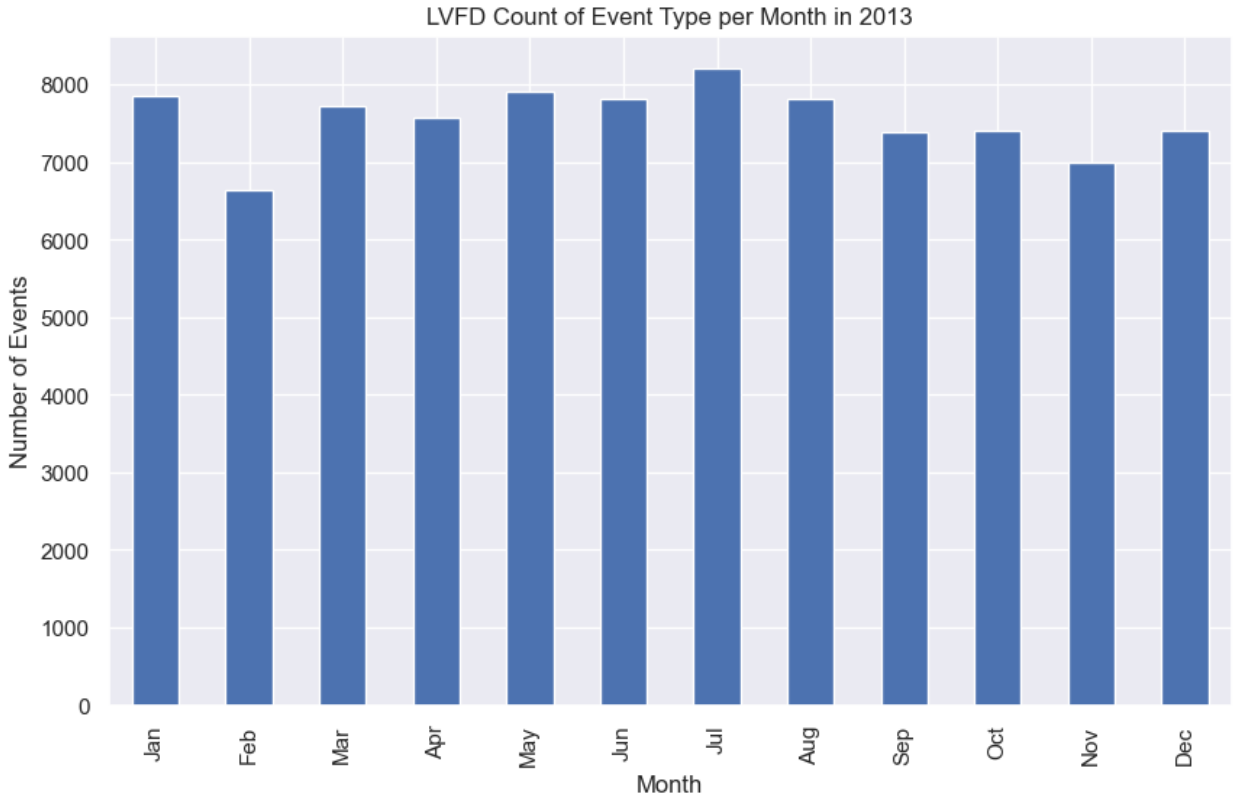

LVFD Count of each Event Types per 2016
100000
80000
60000
40000
20000
0
Number of Events
Medical
Public Need
Fire
Other
Event Type

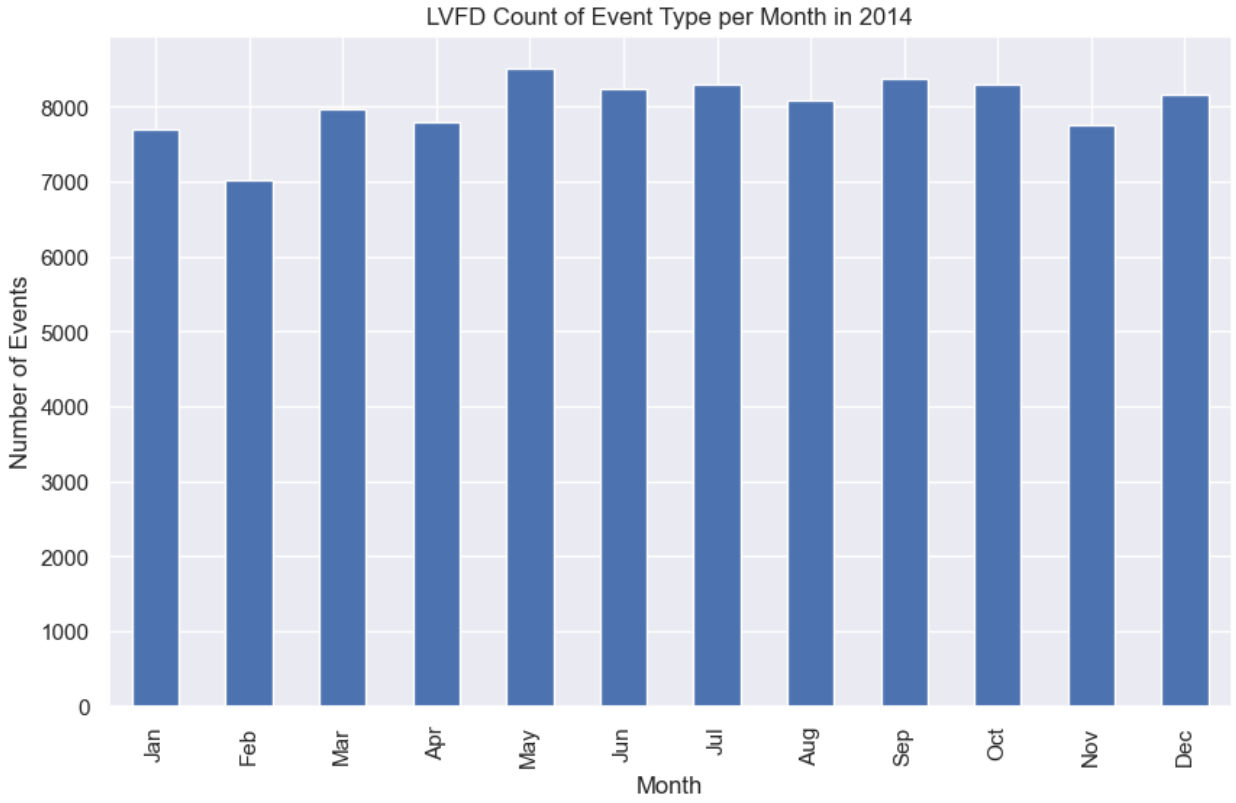

LVFD Count of each Event Types per 2017
Number of Events
80000
60000
40000
20000
0
Medical
Public Need
Fire
Other
Event Type

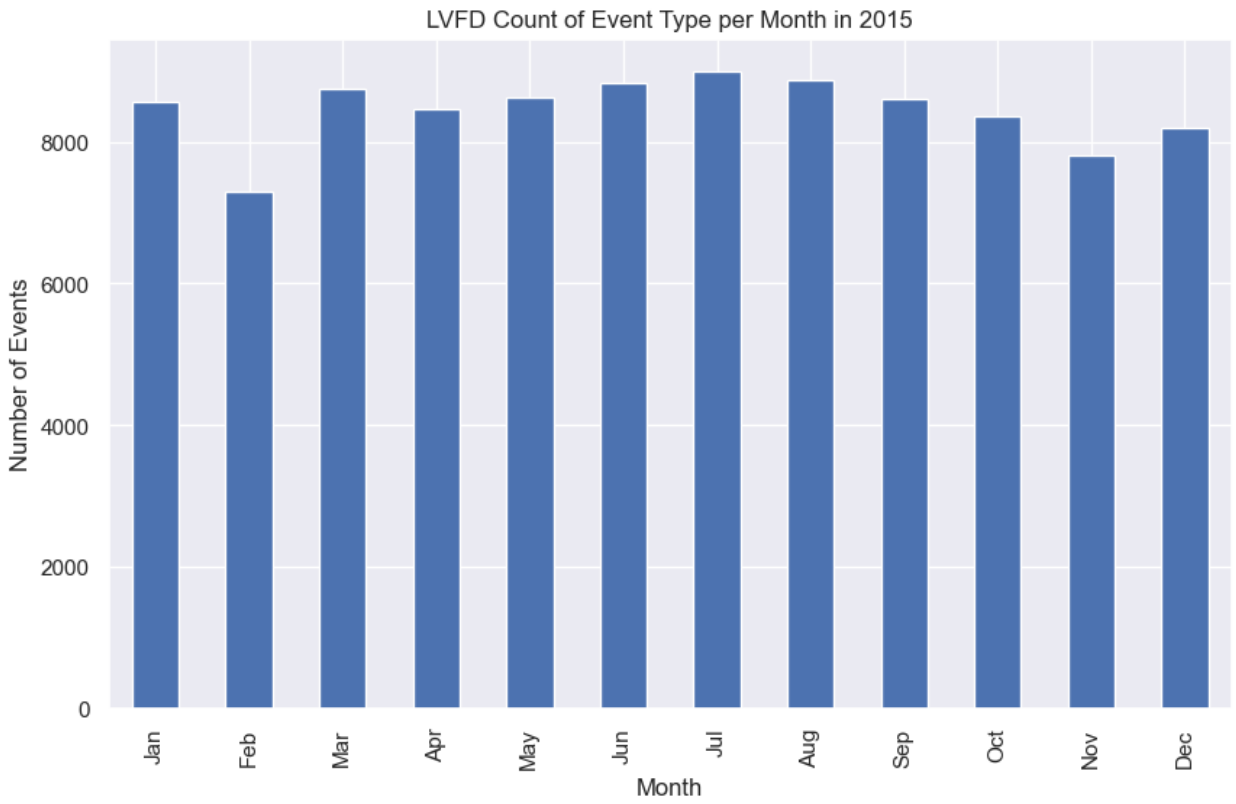


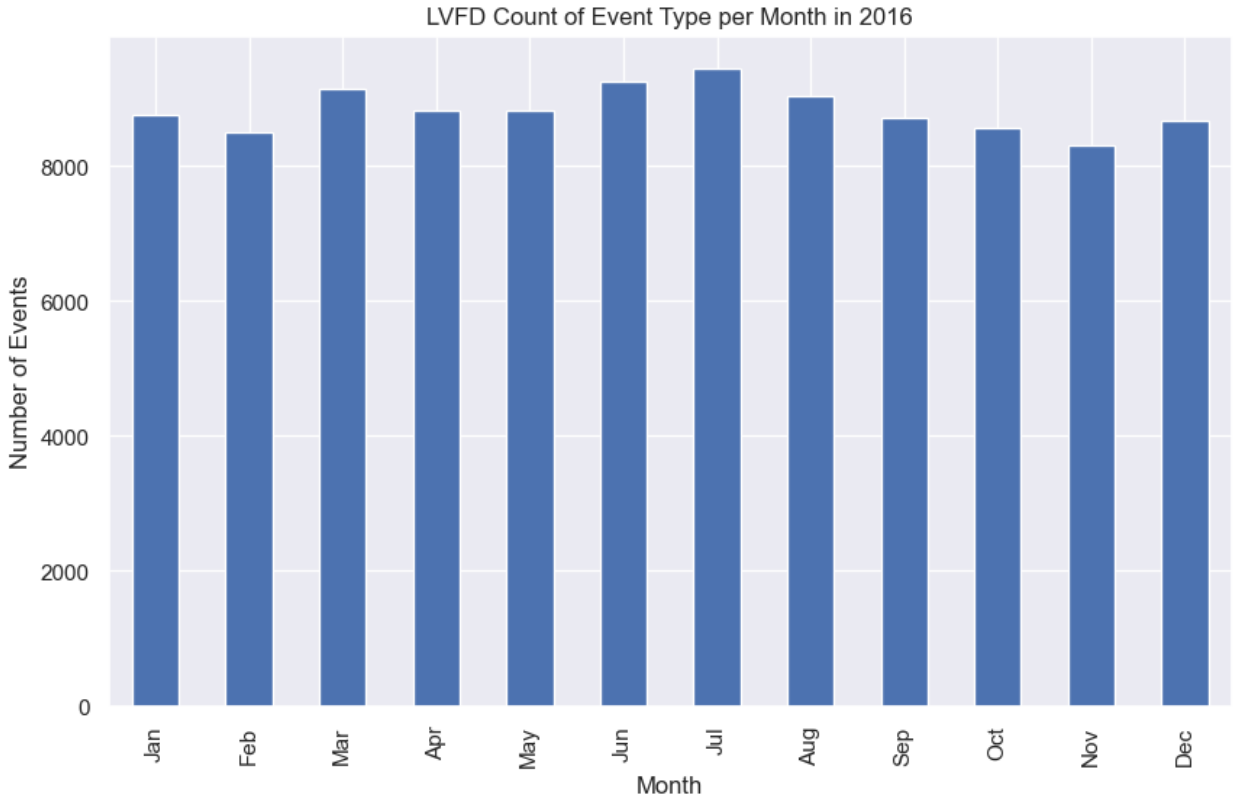

```
# Create a plot for each year
for year in years:
    events_by_month_year = df[df['Year_Extracted'] == year]
['Month_Extracted'].value_counts().sort_index()

    # Plot the data
    plt.figure(figsize=(10, 6))
    events_by_month_year.plot(kind='bar')
    plt.xlabel('Month')
    plt.ylabel('Number of Events')
    plt.title(f'LVFD Count of Event Type per Month in {year}')
    plt.xticks(ticks=range(0, 12), labels=['Jan', 'Feb', 'Mar', 'Apr',
'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])
    plt.show()
```

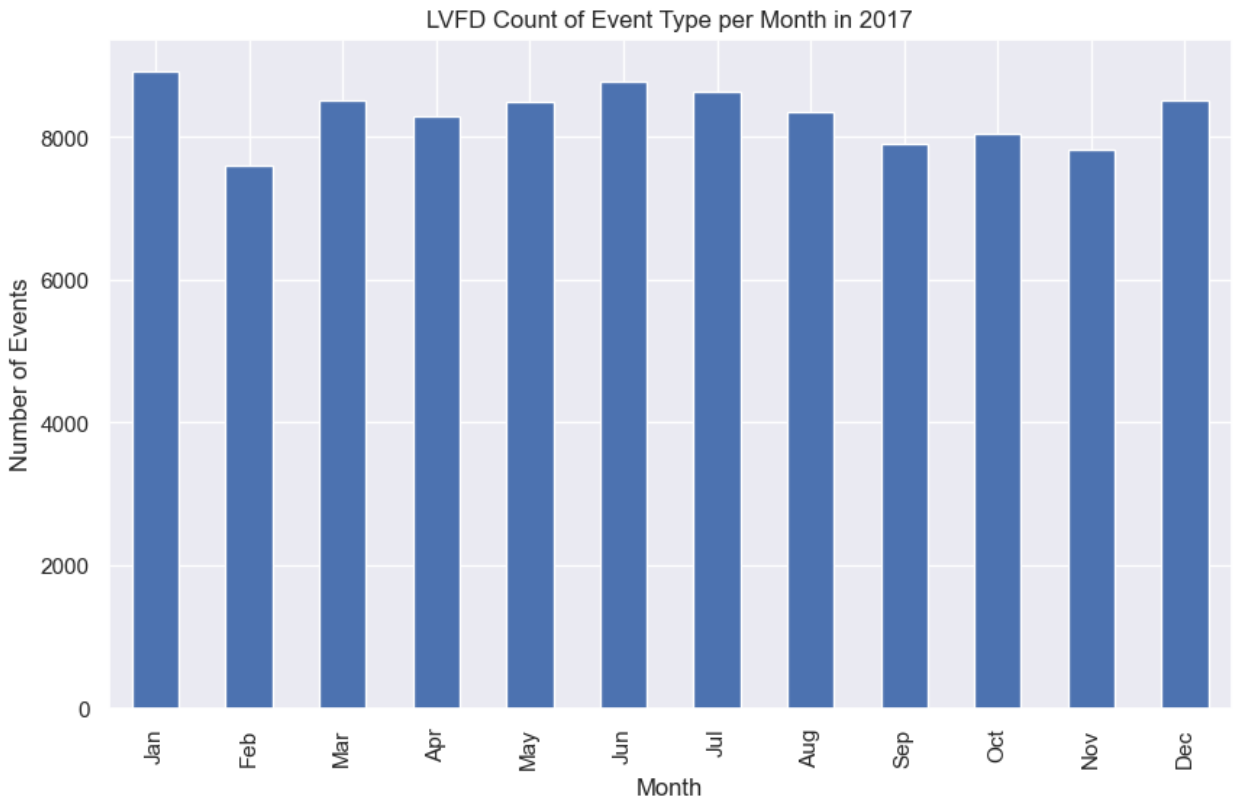

LVFD Count of Event Type per Month in 2012
Number of Events
0
1000
2000
3000
4000
5000
6000
7000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

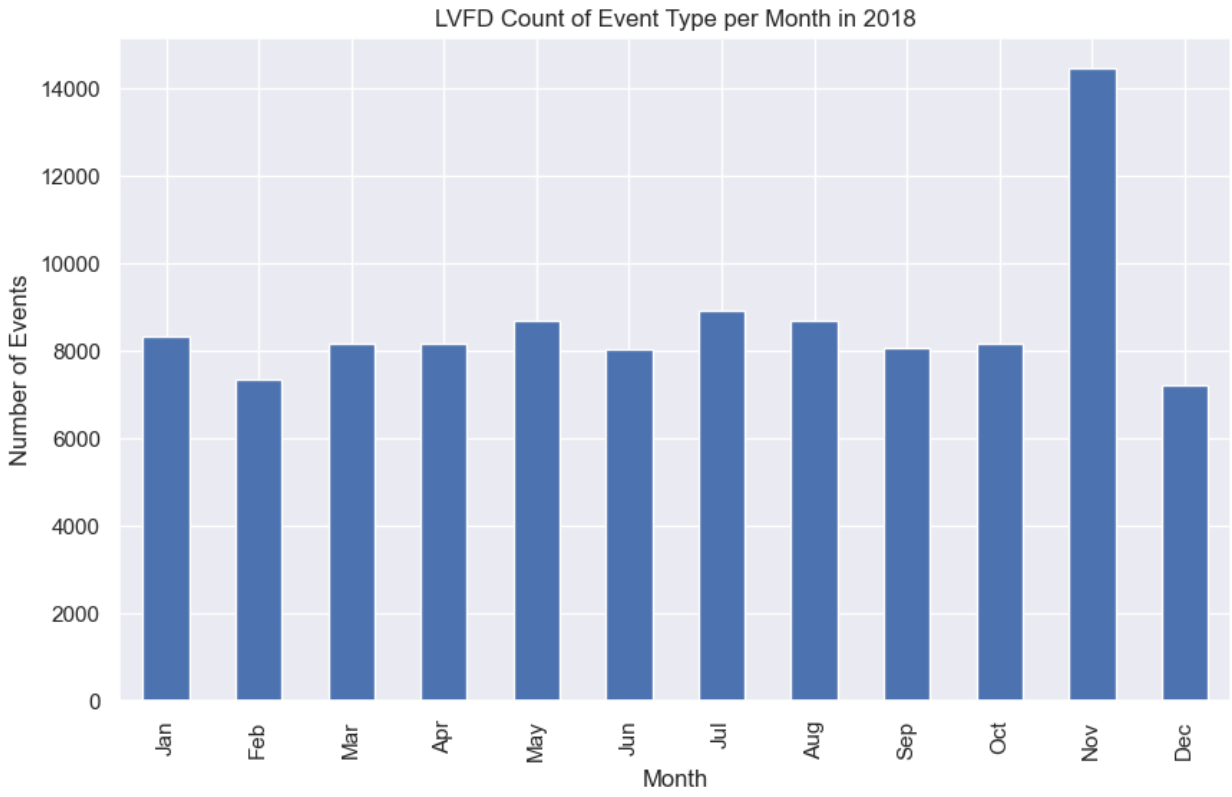

LVFD Count of Event Type per Month in 2013
Number of Events
0
1000
2000
3000
4000
5000
6000
7000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


LVFD Count of Event Type per Month in 2014
Number of Events
0
1000
2000
3000
4000
5000
6000
7000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


LVFD Count of Event Type per Month in 2015
Number of Events
0
2000
4000
6000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

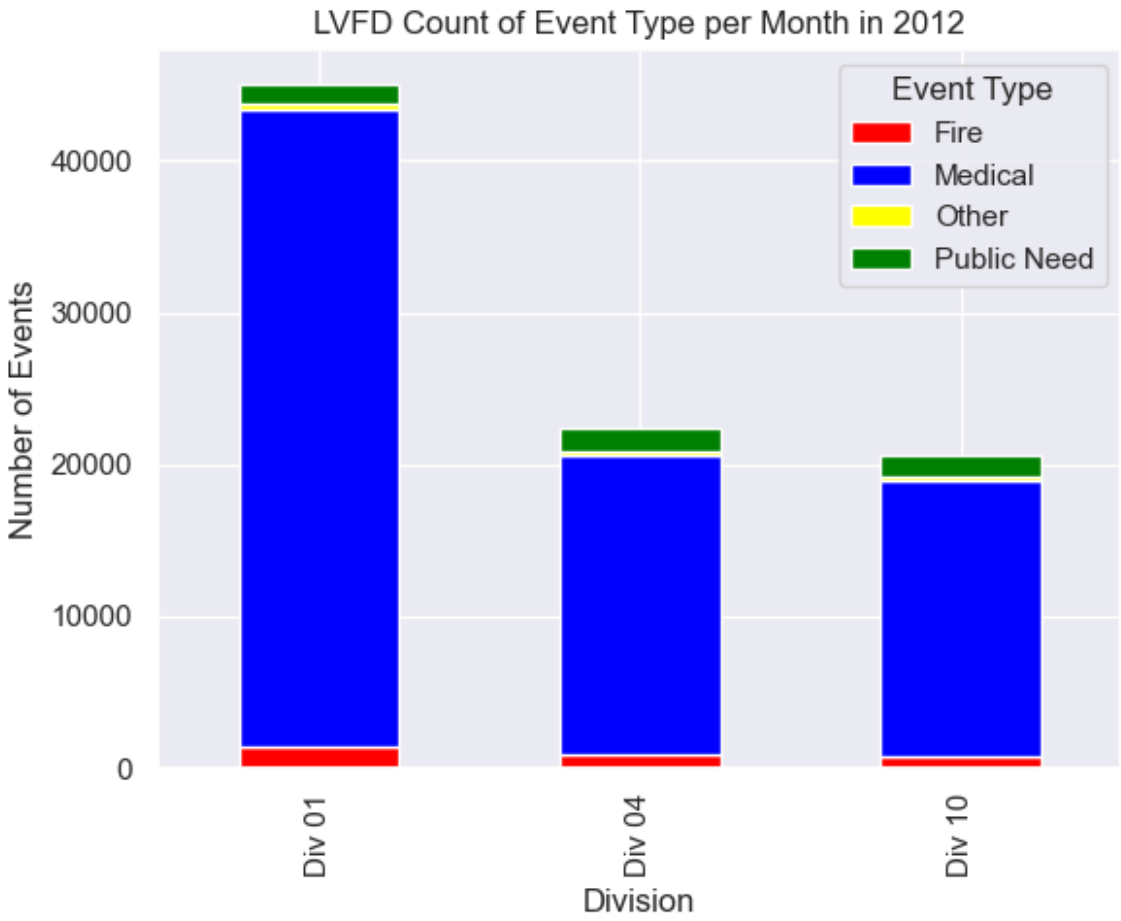

LVFD Count of Event Type per Month in 2016
Number of Events
0
2000
4000
6000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


LVFD Count of Event Type per Month in 2017
Number of Events
0
2000
4000
6000
8000
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month

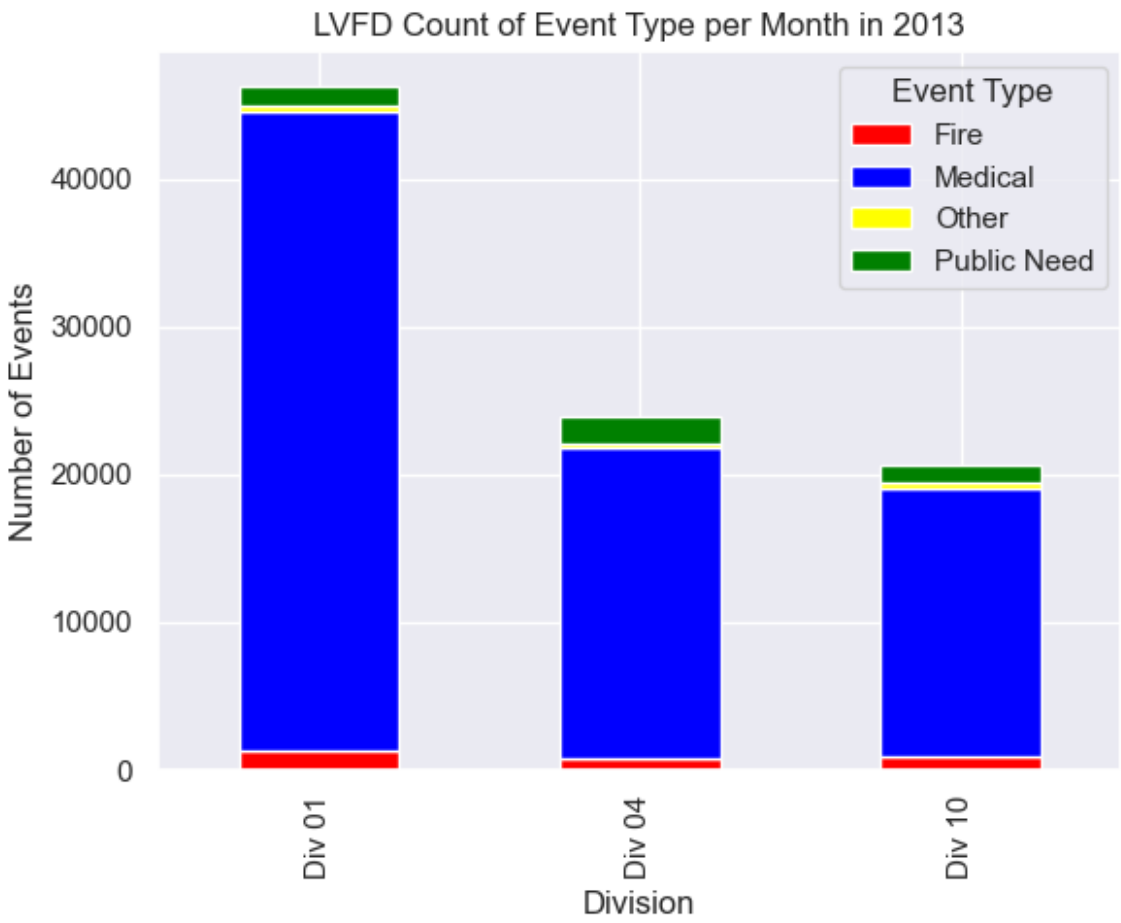



```
# Create a plot for each year
for year in years:
    events_by_division_year = df[df['Year_Extracted'] ==
year].groupby(['Division', 'Event_Type']).size().unstack().fillna(0)

    # Map colors to event types
    colors = [color_scheme.get(event_type, 'black') for event_type in
events_by_division_year.columns]

    # Plot the data
    plt.figure(figsize=(10, 6))
    events_by_division_year.plot(kind='bar', stacked=True,
color=colors)
    plt.xlabel('Division')
    plt.ylabel('Number of Events')
    plt.title(f'LVFD Count of Event Type per Month in {year}')
    plt.legend(title='Event Type')
    plt.show()
```

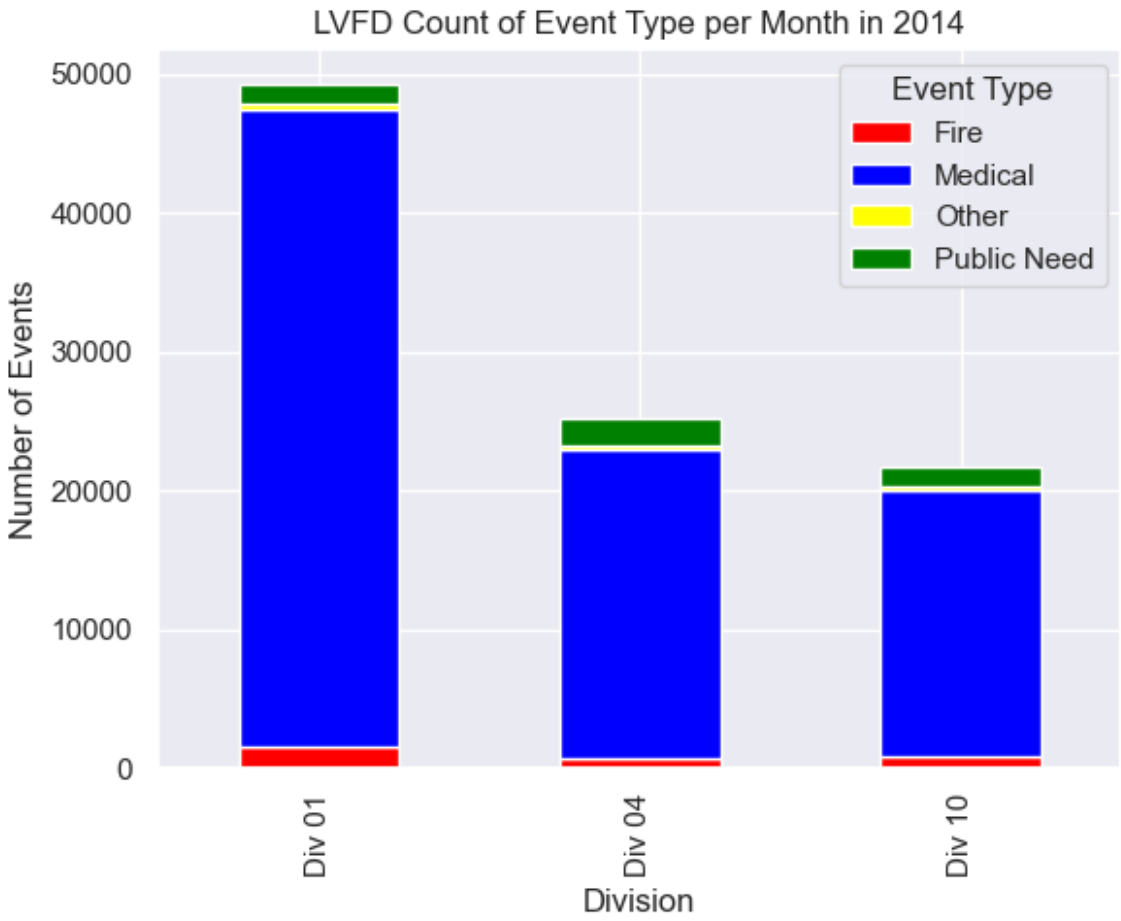
```
<Figure size 1000x600 with 0 Axes>
```



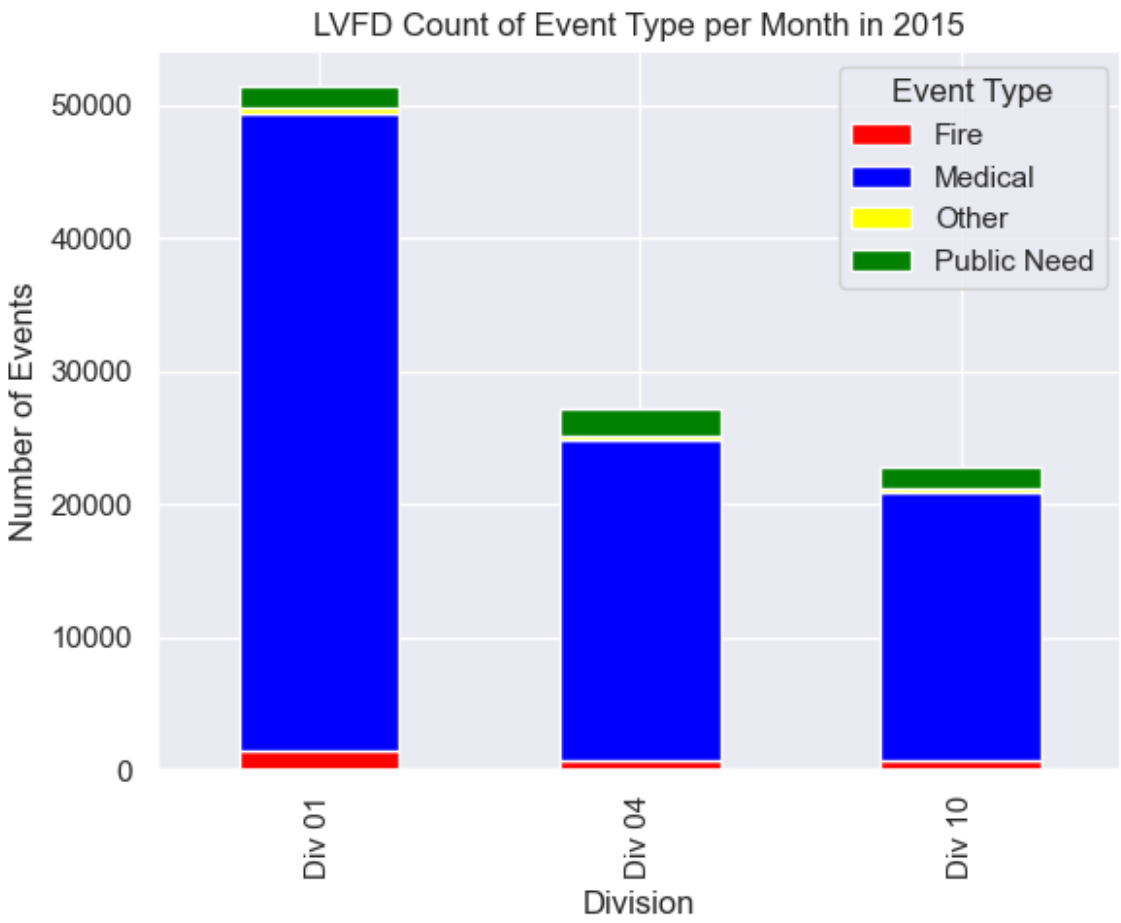

```
<Figure size 1000x600 with 0 Axes>
```



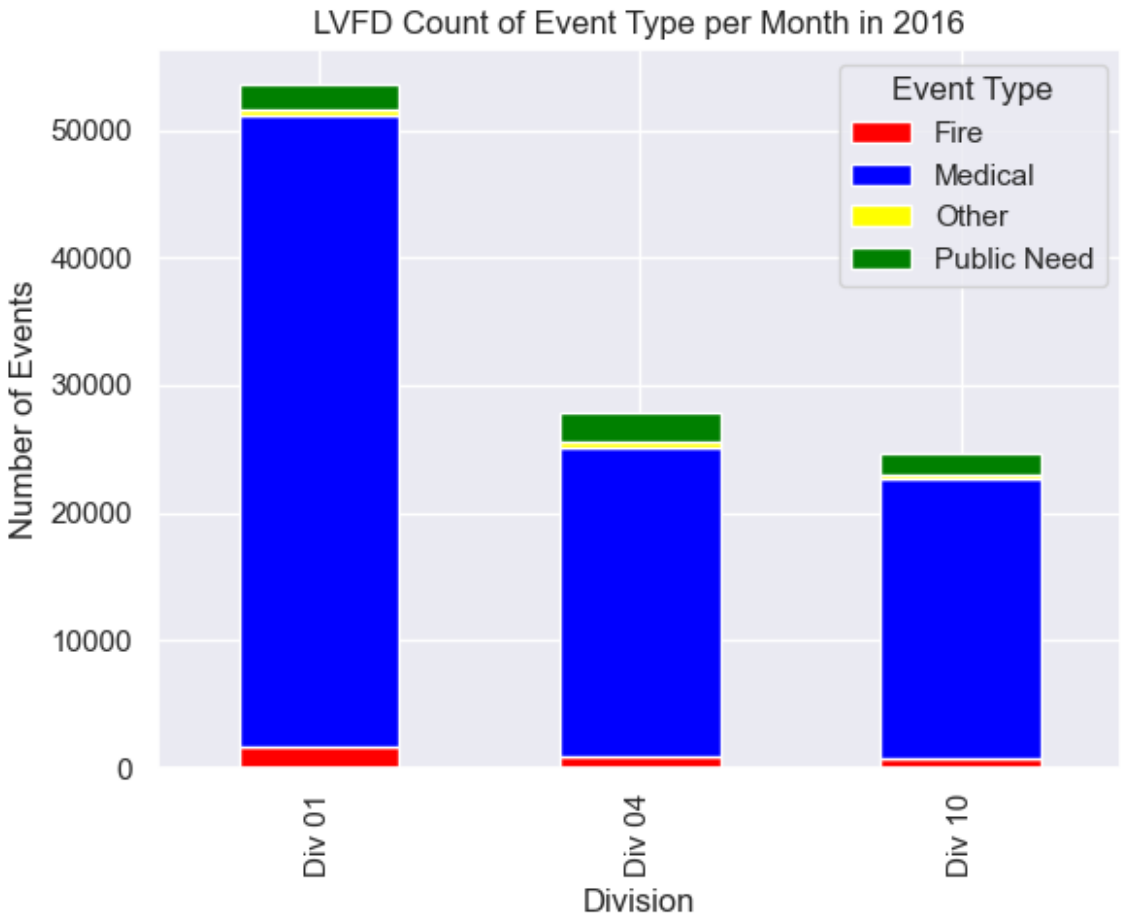

```
<Figure size 1000x600 with 0 Axes>
```



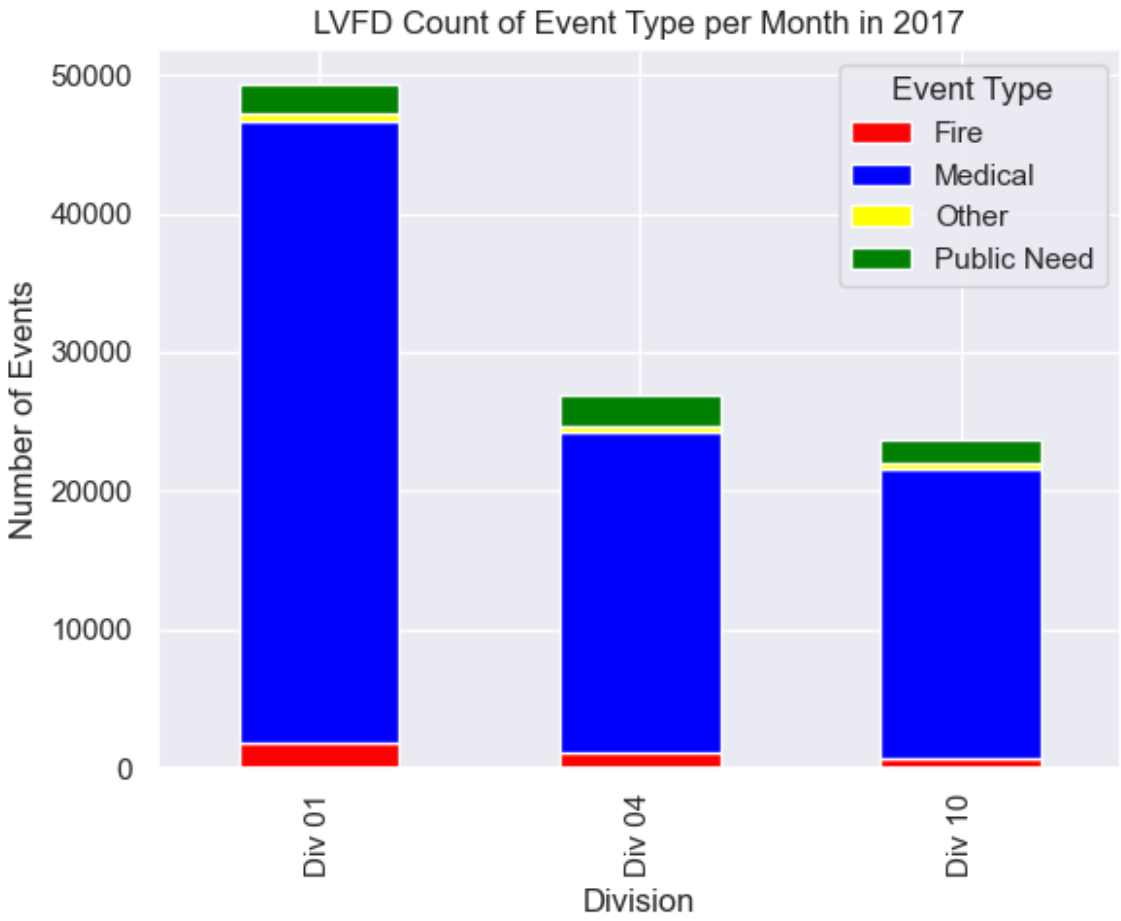

```
<Figure size 1000x600 with 0 Axes>
```



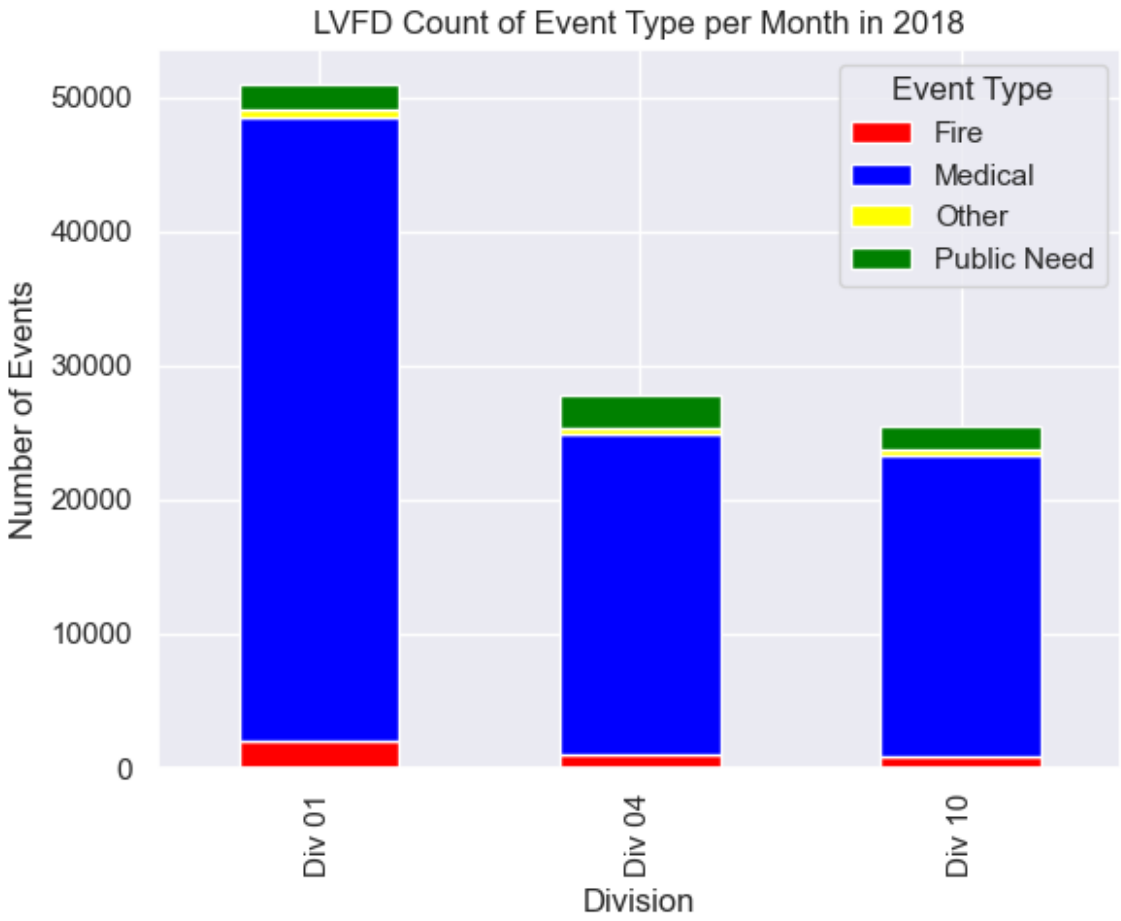

```
<Figure size 1000x600 with 0 Axes>
```



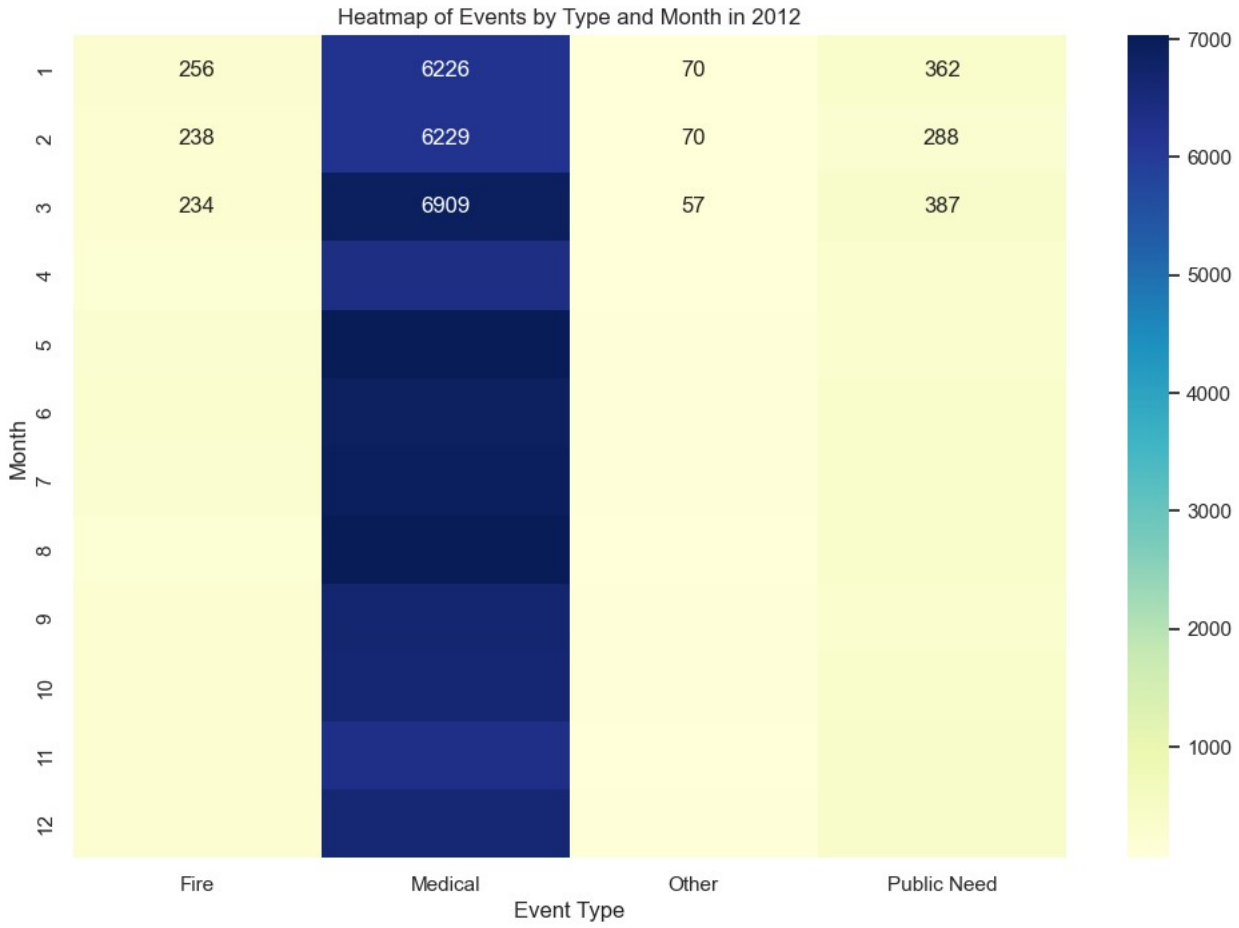

```
<Figure size 1000x600 with 0 Axes>
```

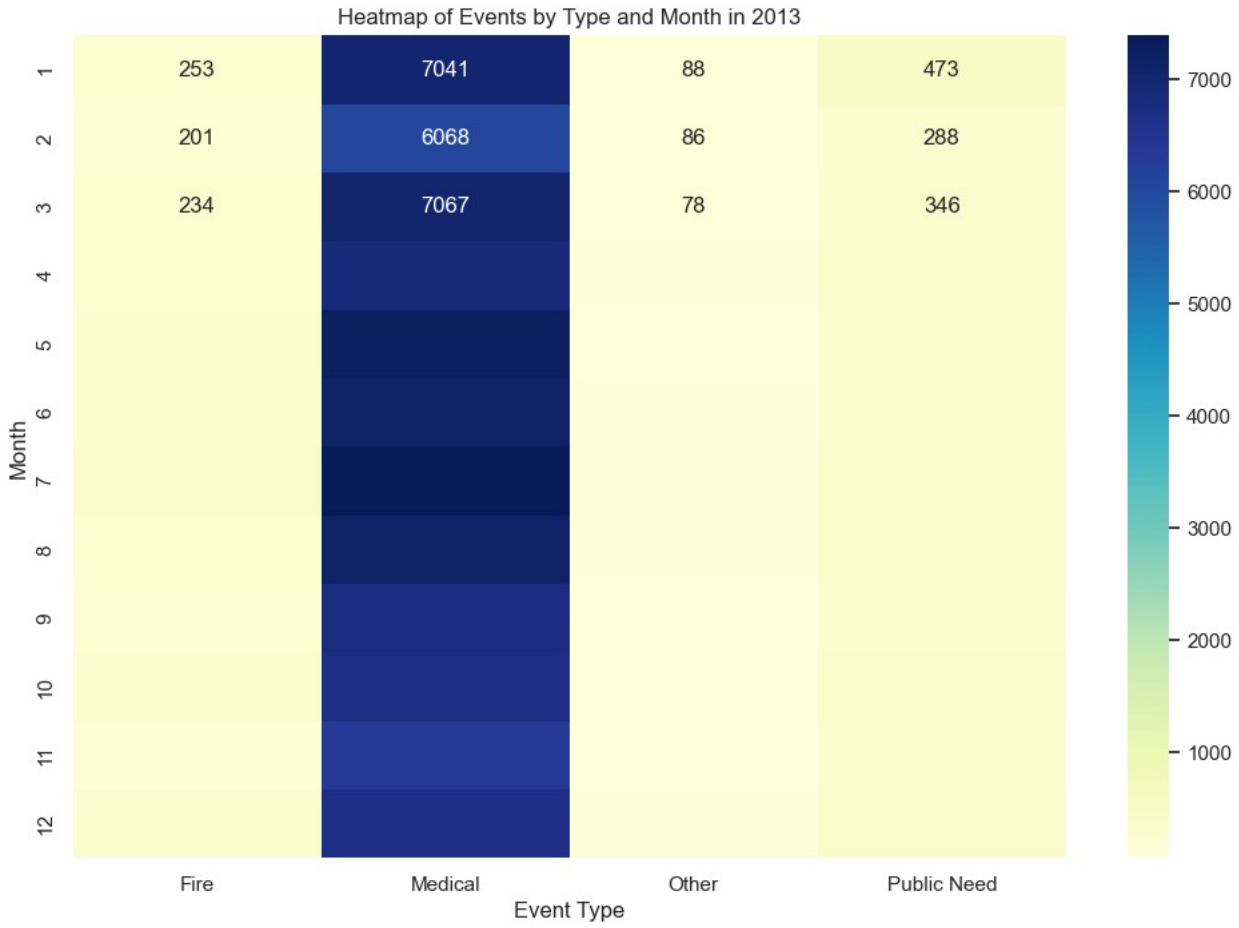


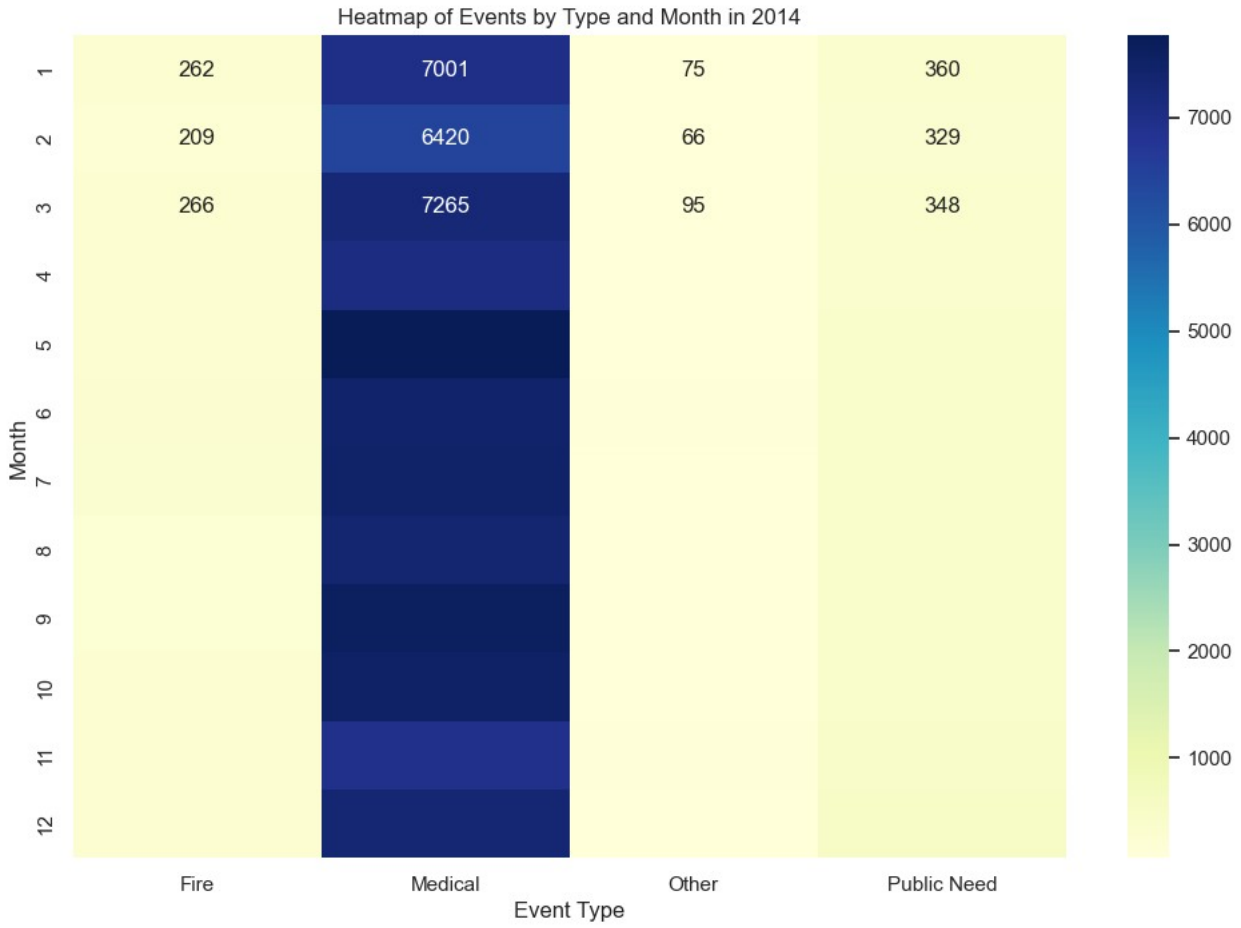

```
<Figure size 1000x600 with 0 Axes>
```

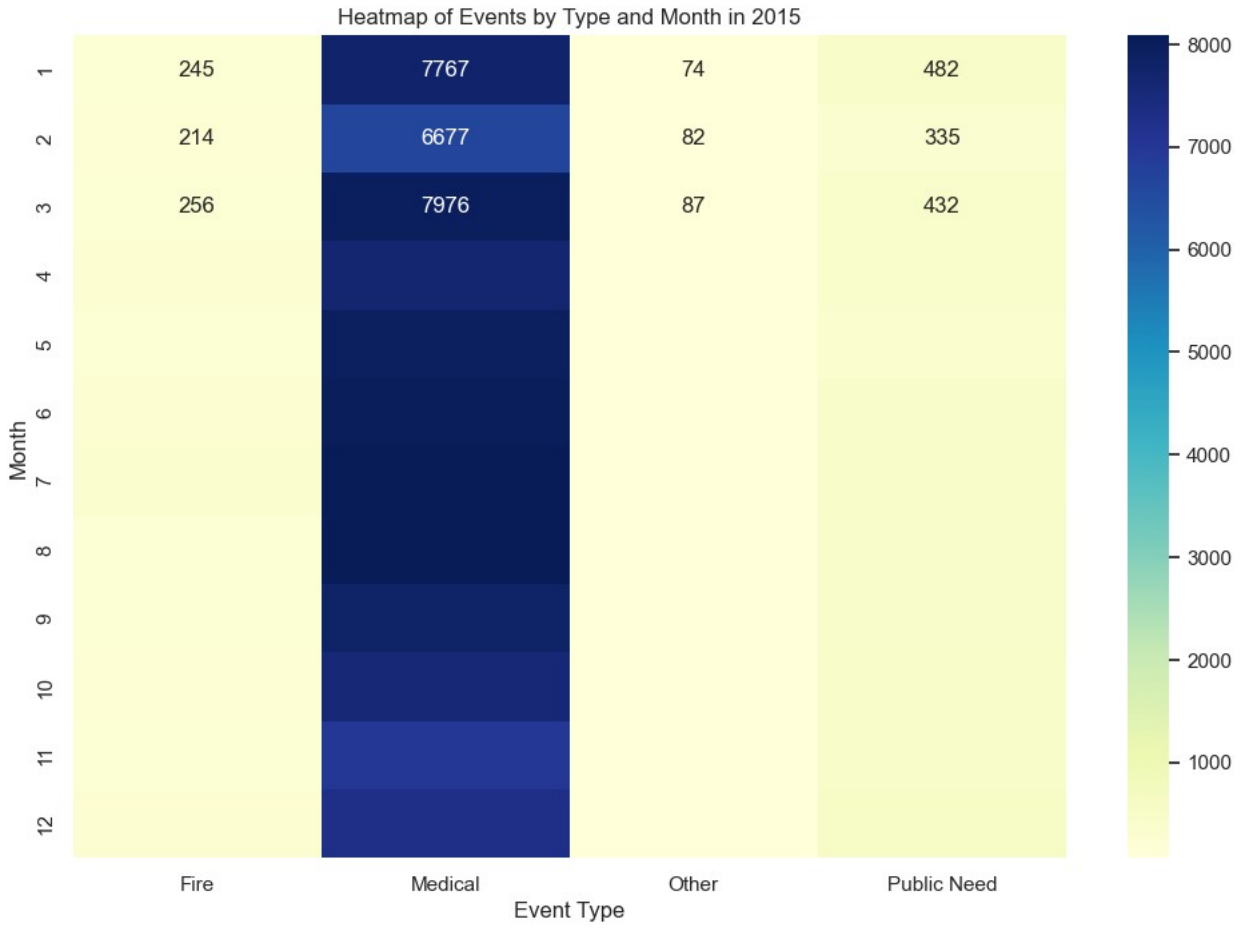


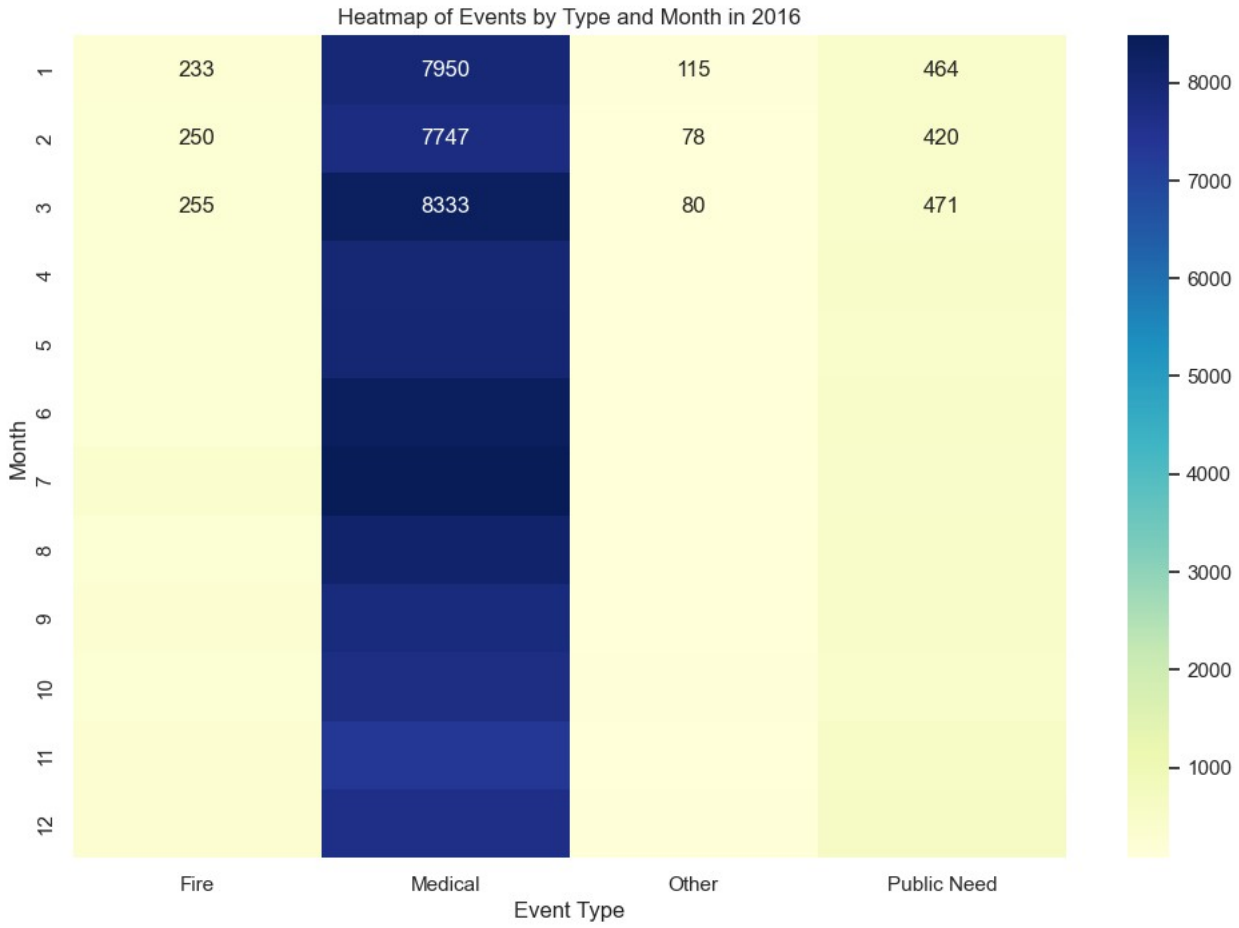

```
import seaborn as sns

# Create a heatmap for each year
for year in years:
    events_by_month_type_year = df[df['Year_Extracted'] ==
year].groupby(['Month_Extracted',
'Event_Type']).size().unstack().fillna(0)

    # Plot the heatmap
    plt.figure(figsize=(12, 8))
    sns.heatmap(events_by_month_type_year, annot=True, fmt="d",
cmap="YlGnBu", cbar=True)
    plt.xlabel('Event Type')
    plt.ylabel('Month')
    plt.title(f'Heatmap of Events by Type and Month in {year}')
    plt.show()
```


Heatmap of Events by Type and Month in 2012
Month
1
2
3
4
5
6
7
8
9
10
11
12
256
6226
70
362
238
6229
70
288
234
6909
57
387
Fire
Medical
Other
Public Need
Event Type
7000
6000
5000
4000
3000
2000
1000

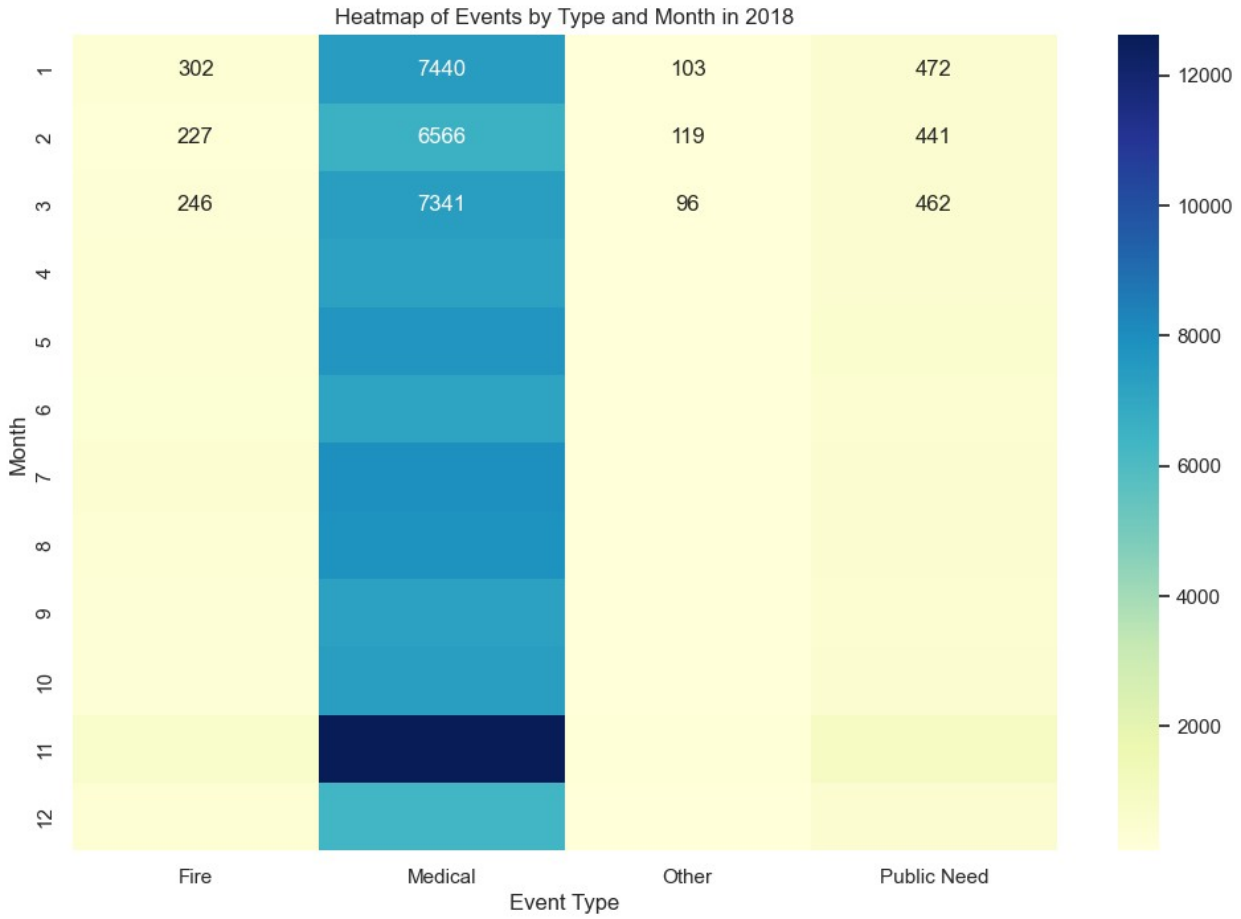

Heatmap of Events by Type and Month in 2013
Month
1
2
3
4
5
6
7
8
9
10
11
12
253
7041
88
473
201
6068
86
288
234
7067
78
346
Fire
Medical
Other
Public Need
Event Type
7000
6000
5000
4000
3000
2000
1000


Heatmap of Events by Type and Month in 2014
Month
1
2
3
4
5
6
7
8
9
10
11
12
262
7001
75
360
209
6420
66
329
266
7265
95
348
Fire
Medical
Other
Public Need
Event Type
7000
6000
5000
4000
3000
2000
1000


Heatmap of Events by Type and Month in 2015
Month
1
2
3
4
5
6
7
8
9
10
11
12
245
7767
74
482
214
6677
82
335
256
7976
87
432
Fire
Medical
Other
Public Need
Event Type
8000
7000
6000
5000
4000
3000
2000
1000


Heatmap of Events by Type and Month in 2016
Month
1
2
3
4
5
6
7
8
9
10
11
12
233
7950
115
464
250
7747
78
420
255
8333
80
471
Fire
Medical
Other
Public Need
Event Type
8000
7000
6000
5000
4000
3000
2000
1000

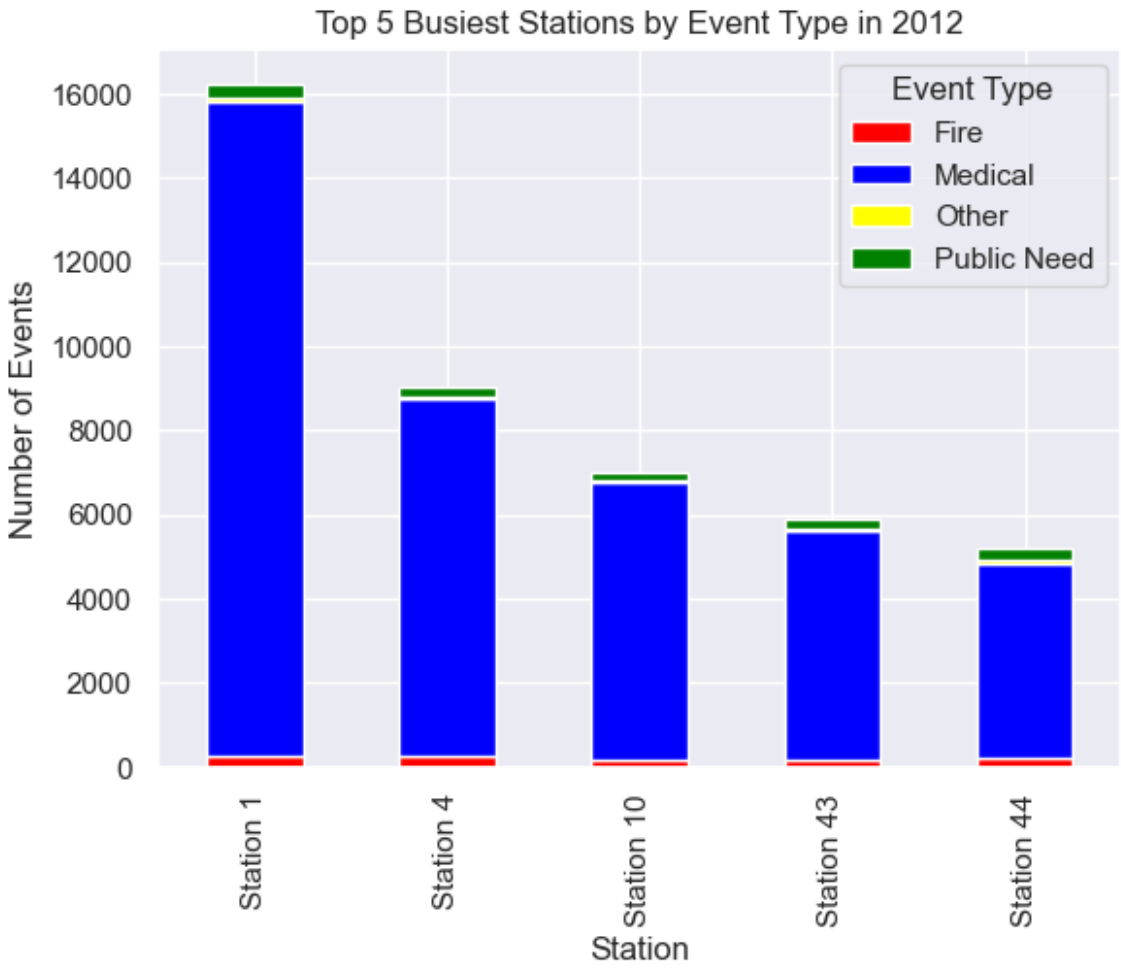

Heatmap of Events by Type and Month in 2017
Month
1
2
3
4
5
6
7
8
9
10
11
12
248
8017
95
569
189
6870
86
456
311
7596
97
519
Fire
Medical
Other
Public Need
Event Type
8000
7000
6000
5000
4000
3000
2000
1000



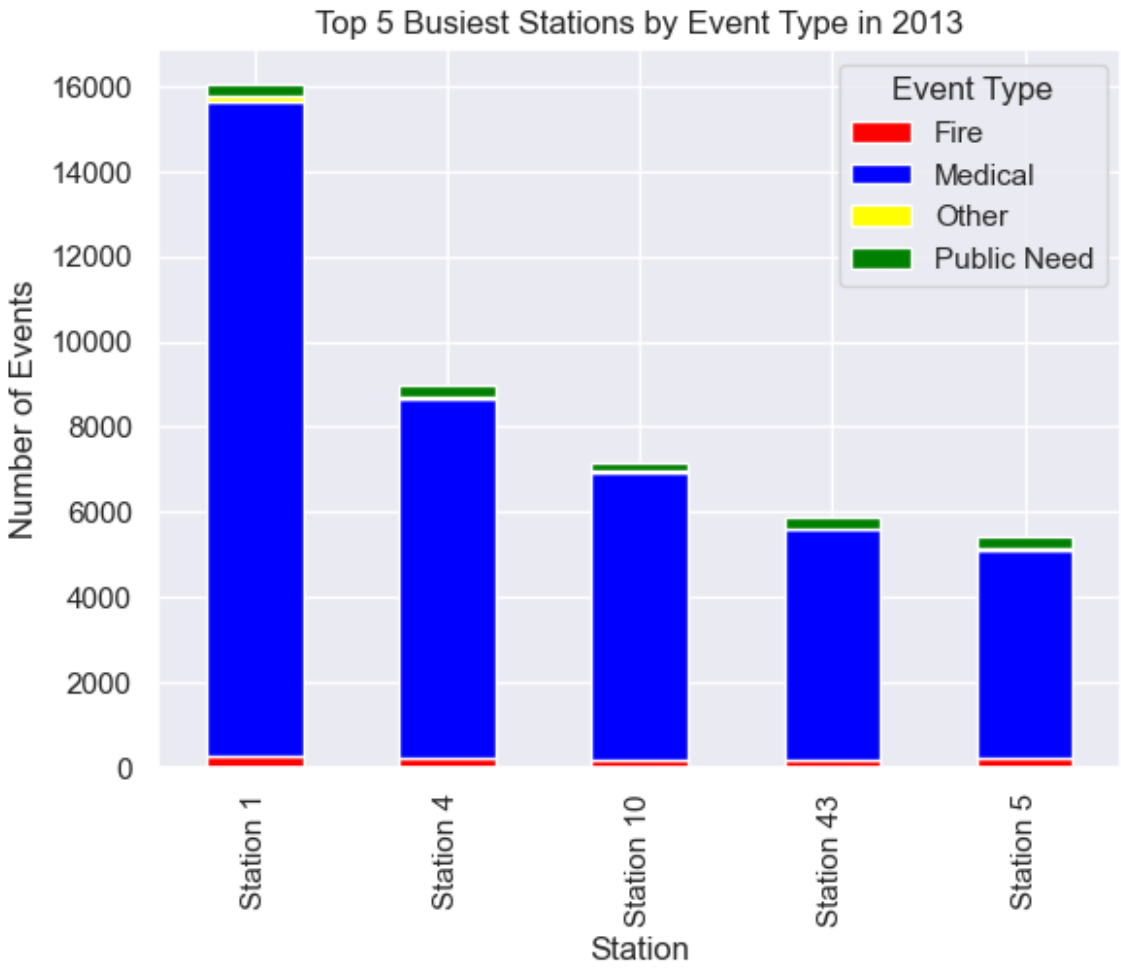

```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming df is your DataFrame
# Define the color scheme for event types
color_scheme = {
    "Fire": "red",
    "Medical": "blue",
    "Public Need": "green",
    "Other": "yellow"
}

# Get a list of unique years
years = df['Year_Extracted'].unique()

# Create a plot for the top 5 busiest stations for each year
for year in years:
    # Filter data for the specific year
    df_year = df[df['Year_Extracted'] == year]

    # Group by station and event type, then count the number of events
    events_by_station_type = df_year.groupby(['Station',
```

```
'Event_Type']).size().unstack().fillna(0)

    # Sum the total events for each station
    total_events_by_station = events_by_station_type.sum(axis=1)

    # Get the top 5 busiest stations
    top_5_stations = total_events_by_station.nlargest(5).index

    # Filter the original grouped data to include only the top 5
stations
    top_5_events = events_by_station_type.loc[top_5_stations]

    # Map colors to event types
    colors = [color_scheme.get(event_type, 'black') for event_type in
top_5_events.columns]

    # Plot the data
    plt.figure(figsize=(12, 8))
    top_5_events.plot(kind='bar', stacked=True, color=colors)
    plt.xlabel('Station')
    plt.ylabel('Number of Events')
    plt.title(f'Top 5 Busiest Stations by Event Type in {year}')
    plt.legend(title='Event Type')
    plt.show()
```

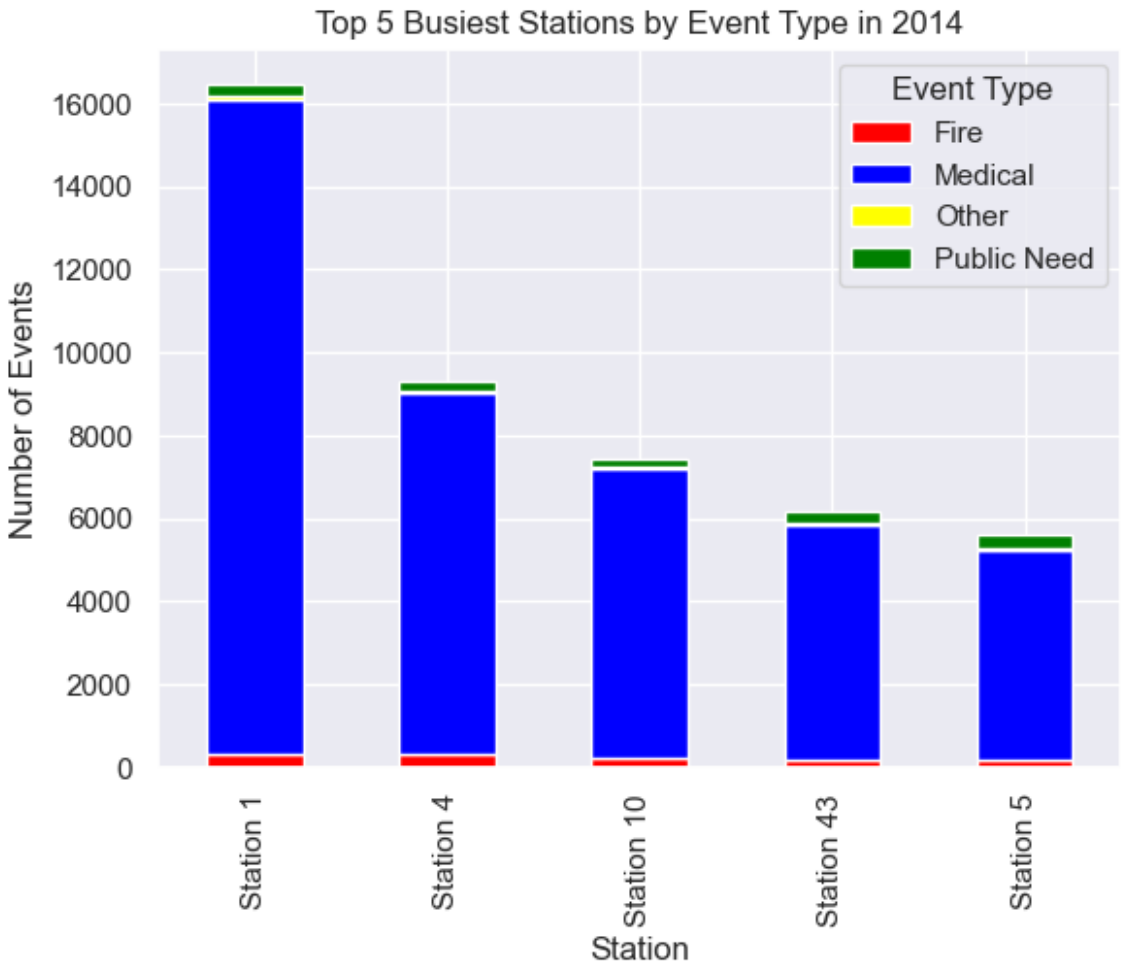
```
<Figure size 1200x800 with 0 Axes>
```



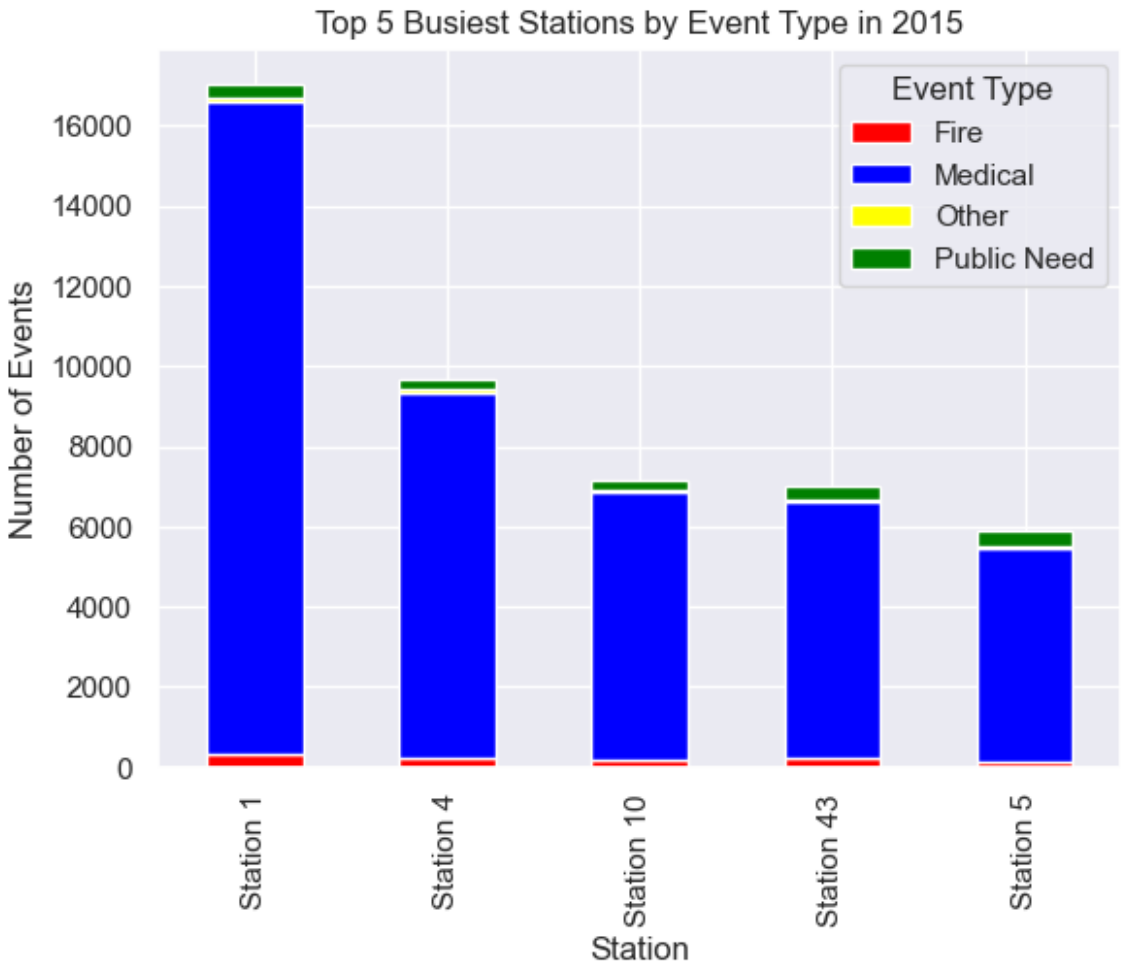

```
<Figure size 1200x800 with 0 Axes>
```



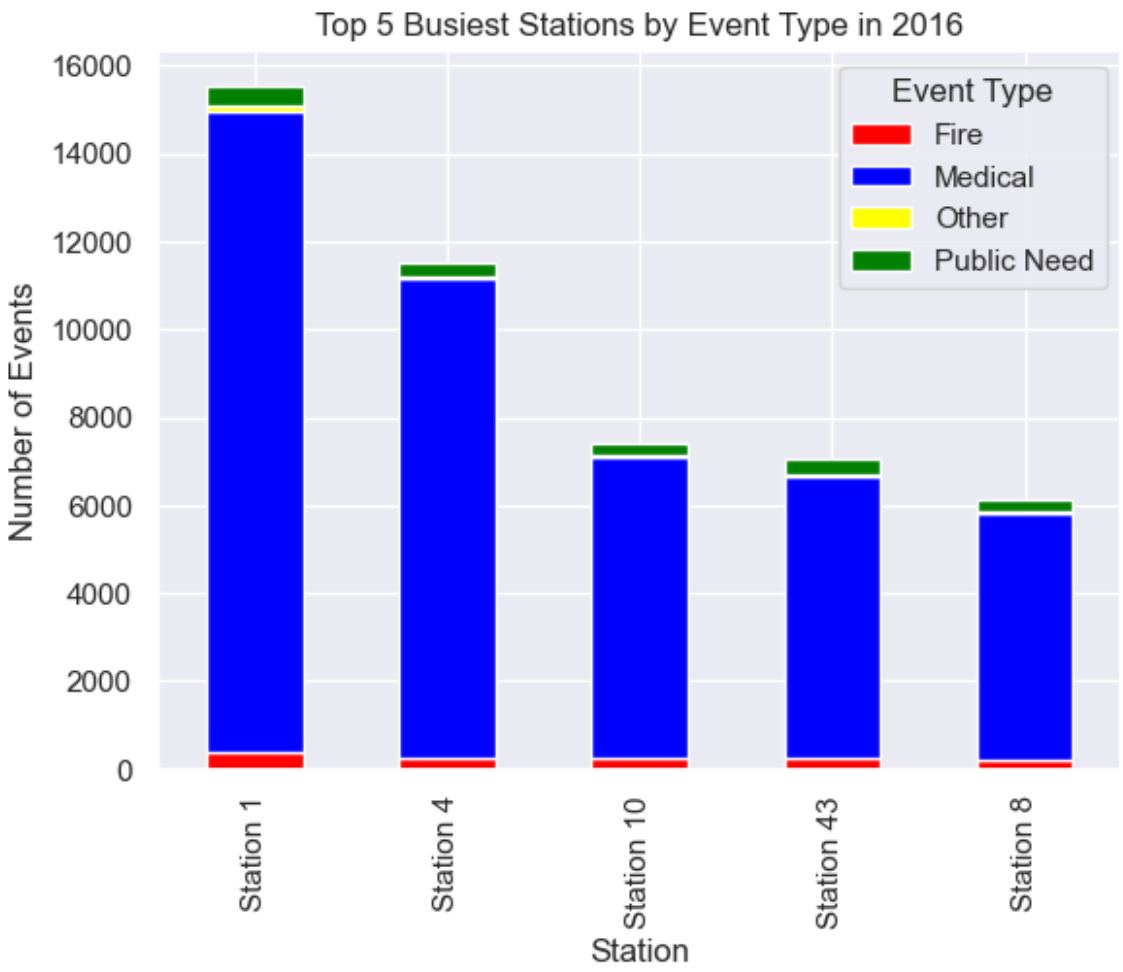

```
<Figure size 1200x800 with 0 Axes>
```



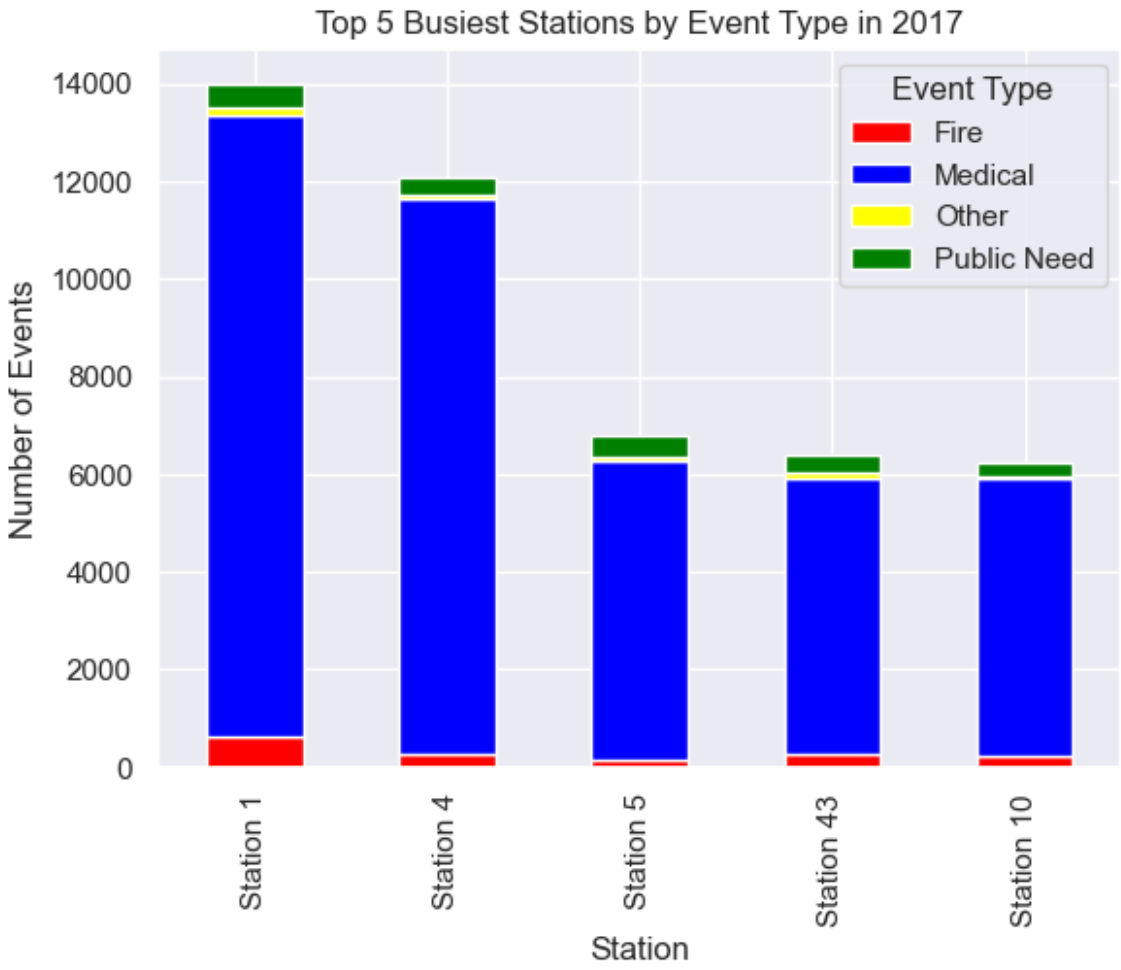

```
<Figure size 1200x800 with 0 Axes>
```



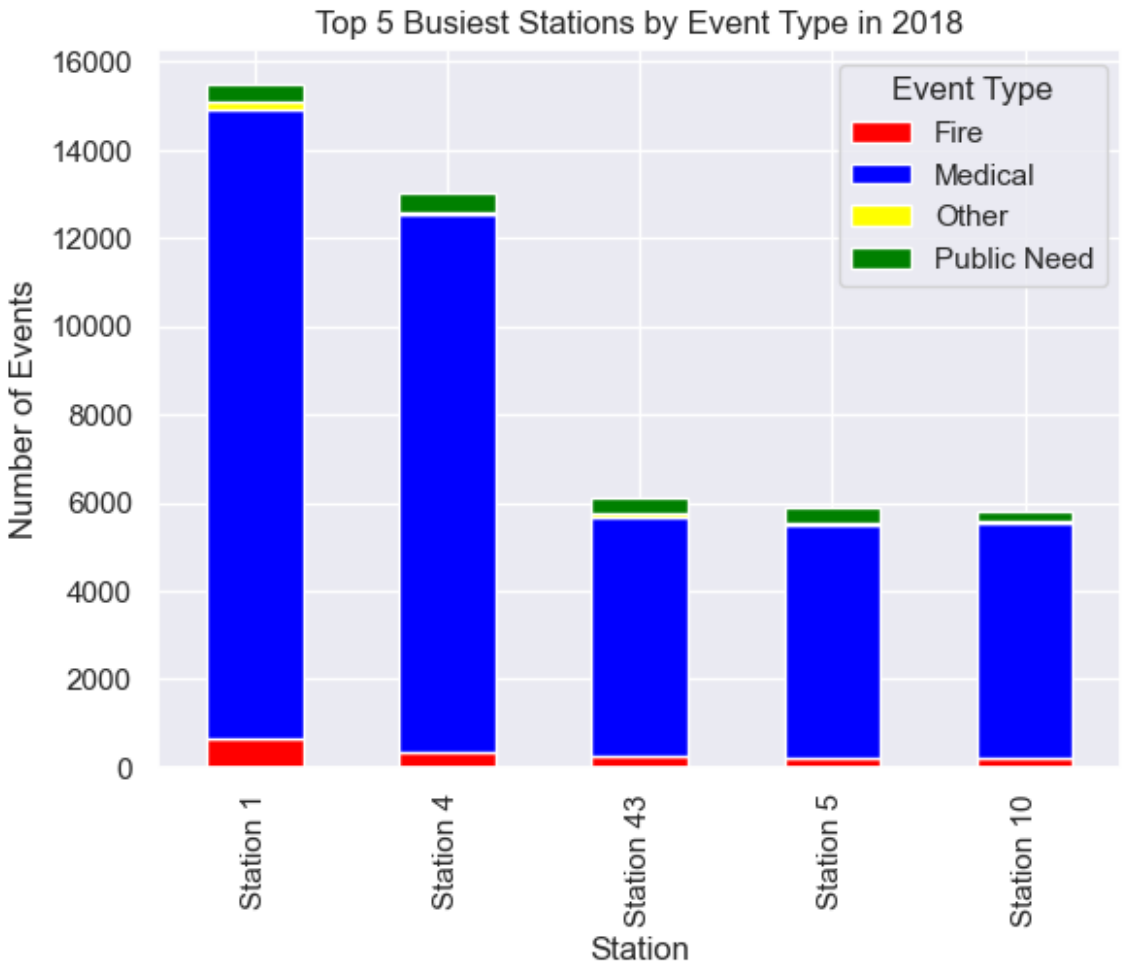

```
<Figure size 1200x800 with 0 Axes>
```




```
<Figure size 1200x800 with 0 Axes>
```




```
<Figure size 1200x800 with 0 Axes>
```




```
# Parse the Location_Coordinates column
df['Location_Coordinates'] =
df['Location_Coordinates'].str.strip('()')
df[['Latitude', 'Longitude']] =
df['Location_Coordinates'].str.split(', ', expand=True).astype(float)

# Group by year and location coordinates, then count the number of
events
events_by_location_year = df.groupby(['Year_Extracted', 'Latitude',
'Longitude']).size().reset_index(name='Events')

# Find the busiest location for each year
busiest_locations =
events_by_location_year.loc[events_by_location_year.groupby('Year_Extr
acted')['Events'].idxmax()]

print(busiest_locations)
```

```
      Year_Extracted   Latitude   Longitude  Events
4759            2012  36.188438 -115.135391     866
```

```
14885            2013  36.188438 -115.135391     790
25161            2014  36.186279 -115.134041     836
36000            2015  36.186279 -115.134041     984
48831            2016  36.186279 -115.134041     818
64077            2017  36.186279 -115.134041     718
77697            2018  36.186279 -115.134041    1140
```

```
import folium

# Parse the Location_Coordinates column
df['Location_Coordinates'] =
df['Location_Coordinates'].str.strip('()')
df[['Latitude', 'Longitude']] =
df['Location_Coordinates'].str.split(', ', expand=True).astype(float)

# Group by year and location coordinates, then count the number of
events
events_by_location_year = df.groupby(['Year_Extracted', 'Latitude',
'Longitude']).size().reset_index(name='Events')

# Find the busiest location for each year
busiest_locations =
events_by_location_year.loc[events_by_location_year.groupby('Year_Extr
acted')['Events'].idxmax()]

# Print the busiest locations for each year
for year in busiest_locations['Year_Extracted'].unique():
    busiest_location_year =
busiest_locations[busiest_locations['Year_Extracted'] == year]
    print(f"Busiest location for {year}: Latitude
{busiest_location_year.iloc[0]['Latitude']}, Longitude
{busiest_location_year.iloc[0]['Longitude']} -
{busiest_location_year.iloc[0]['Events']} events")
```

```
Busiest location for 2012: Latitude 36.18843842, Longitude -
115.1353912 - 866.0 events
Busiest location for 2013: Latitude 36.18843842, Longitude -
115.1353912 - 790.0 events
Busiest location for 2014: Latitude 36.1862793, Longitude -115.1340408
- 836.0 events
Busiest location for 2015: Latitude 36.1862793, Longitude -115.1340408
- 984.0 events
Busiest location for 2016: Latitude 36.1862793, Longitude -115.1340408
- 818.0 events
Busiest location for 2017: Latitude 36.1862792969, Longitude -
115.134040833 - 718.0 events
Busiest location for 2018: Latitude 36.1862792969, Longitude -
115.134040833 - 1140.0 events
```

```python
# Function to find address for a given latitude and longitude
def find_address(latitude, longitude):
    # Search for rows where latitude and longitude match
    matching_rows = df[(df['Latitude'] == latitude) & (df['Longitude'] == longitude)]

    # If there are matching rows, return the address from the first row
    if not matching_rows.empty:
        return matching_rows.iloc[0]['Location']
    else:
        return 'Address not found'

# Print addresses for the identified busiest locations
for location in busiest_locations:
    address = find_address(location["latitude"], location["longitude"])
    print(f"Busiest Location for {location['year']}: Latitude {location['latitude']}, Longitude {location['longitude']} - {location['events']} events - Address: {address}")
```

```
Busiest Location for 2012: Latitude 36.18843842, Longitude -115.1353912 - 866 events - Address: 100 block of W Owens Ave
Busiest Location for 2013: Latitude 36.18843842, Longitude -115.1353912 - 790 events - Address: 100 block of W Owens Ave
Busiest Location for 2014: Latitude 36.1862793, Longitude -115.1340408 - 836 events - Address: 1500 block of N Las Vegas Blvd
Busiest Location for 2015: Latitude 36.1862793, Longitude -115.1340408 - 984 events - Address: 1500 block of N Las Vegas Blvd
Busiest Location for 2016: Latitude 36.1862793, Longitude -115.1340408 - 818 events - Address: 1500 block of N Las Vegas Blvd
Busiest Location for 2017: Latitude 36.1862792969, Longitude -115.134040833 - 718 events - Address: 1500 block of N Las Vegas Blvd
Busiest Location for 2018: Latitude 36.1862792969, Longitude -115.134040833 - 1140 events - Address: 1500 block of N Las Vegas Blvd
```

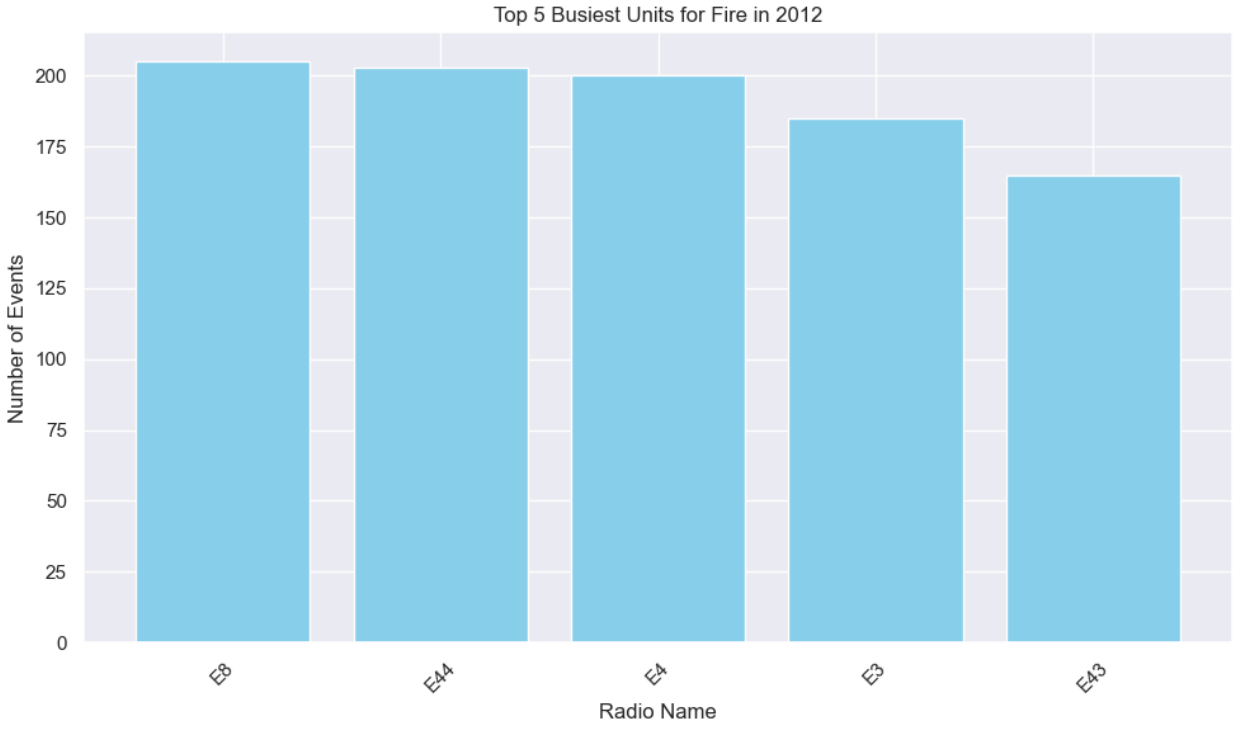
```
df.head()
```

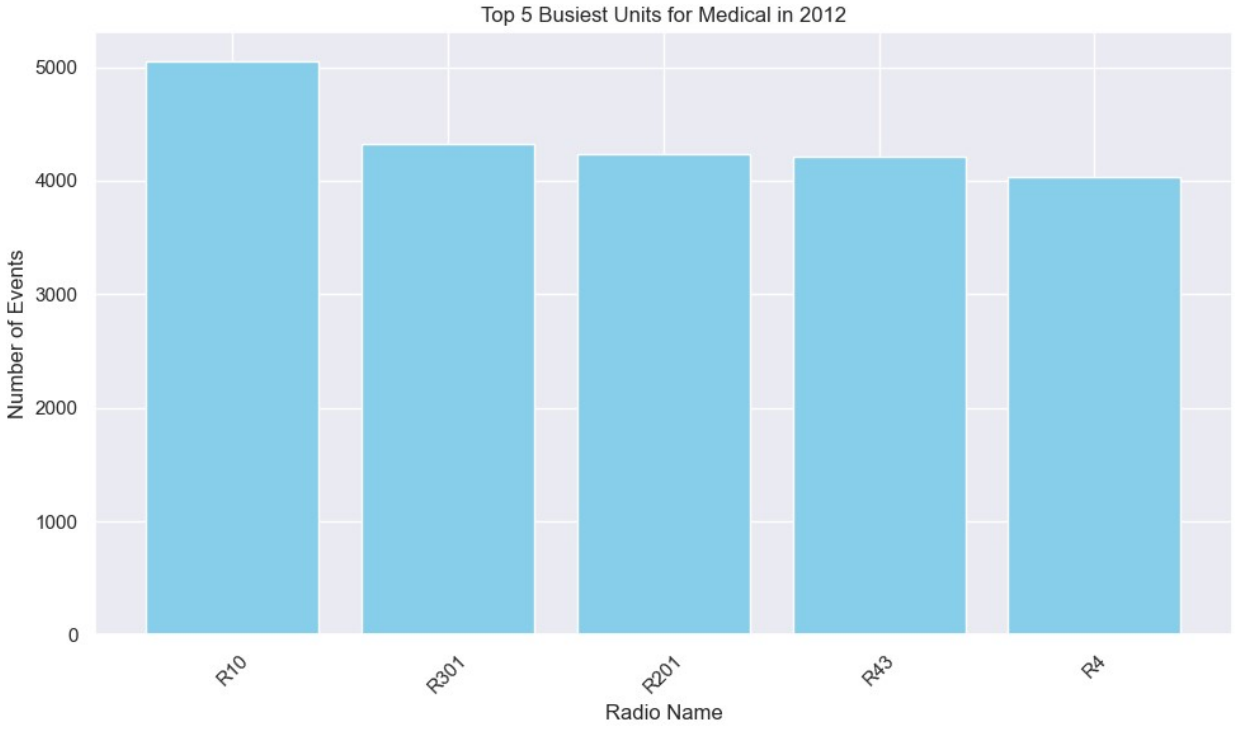
```
              Response_Date  Year_Extracted  Month_Extracted  \
Event_Type
0 2012-01-01 05:30:00+00:00            2012                1
Medical
1 2012-01-01 05:31:00+00:00            2012                1
Medical
2 2012-01-01 05:32:00+00:00            2012                1
Medical
3 2012-01-01 14:57:00+00:00            2012                1
Medical
4 2012-01-01 14:58:00+00:00            2012                1
Medical
```

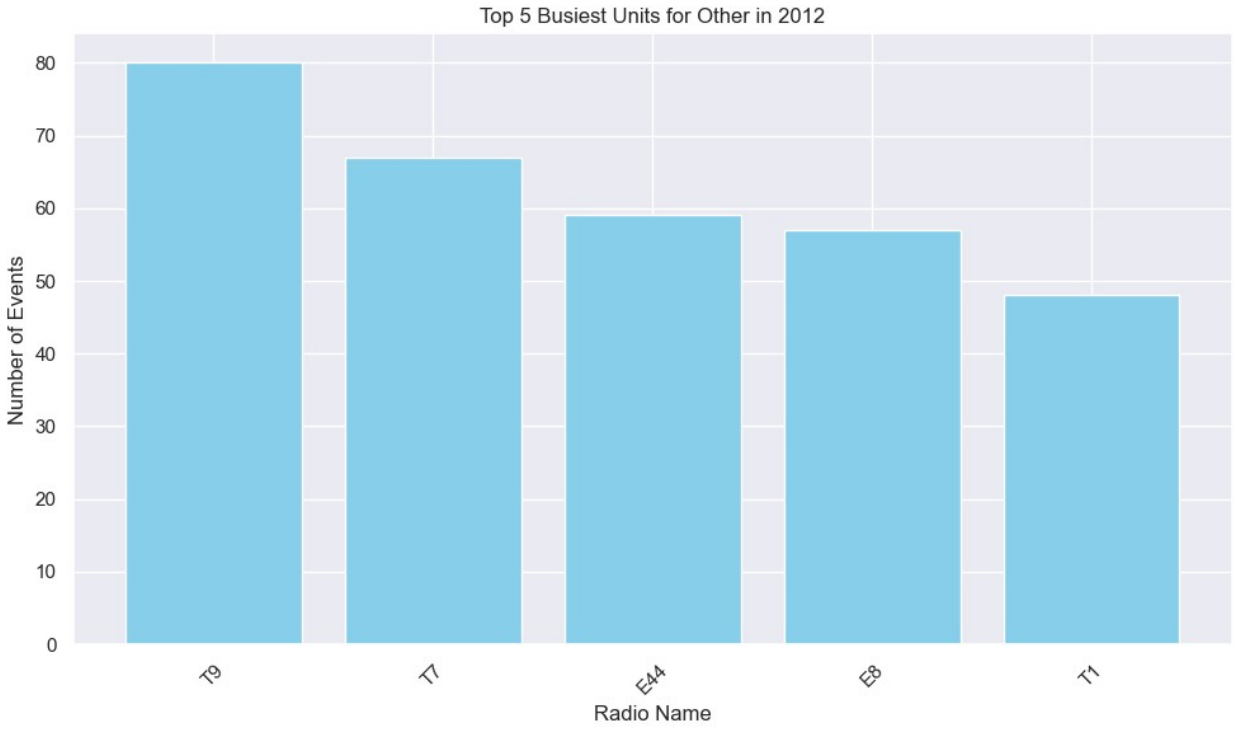
```
  Division     Station Radio_Name                         Location  \
0   Div 01   Station 1       R201        100 block of FREMONT ST
1   Div 01   Station 3         R3        900 block of Pyramid Dr
2   Div 10   Station 2         R2          9000 block of Alta Dr
3   Div 01  Station 10        R10  Cordova St  &  E St Louis Ave
4   Div 10  Station 44        E44      6200 block of Clarice Ave

        Location_Coordinates Year_Month   Latitude   Longitude
0   36.17168427, -115.146347    2012-01  36.171684 -115.146347
1  36.18119431, -115.1897278    2012-01  36.181194 -115.189728
2  36.16799927, -115.2944489    2012-01  36.167999 -115.294449
3  36.14778519, -115.1427994    2012-01  36.147785 -115.142799
4  36.17484665, -115.2262726    2012-01  36.174847 -115.226273
```

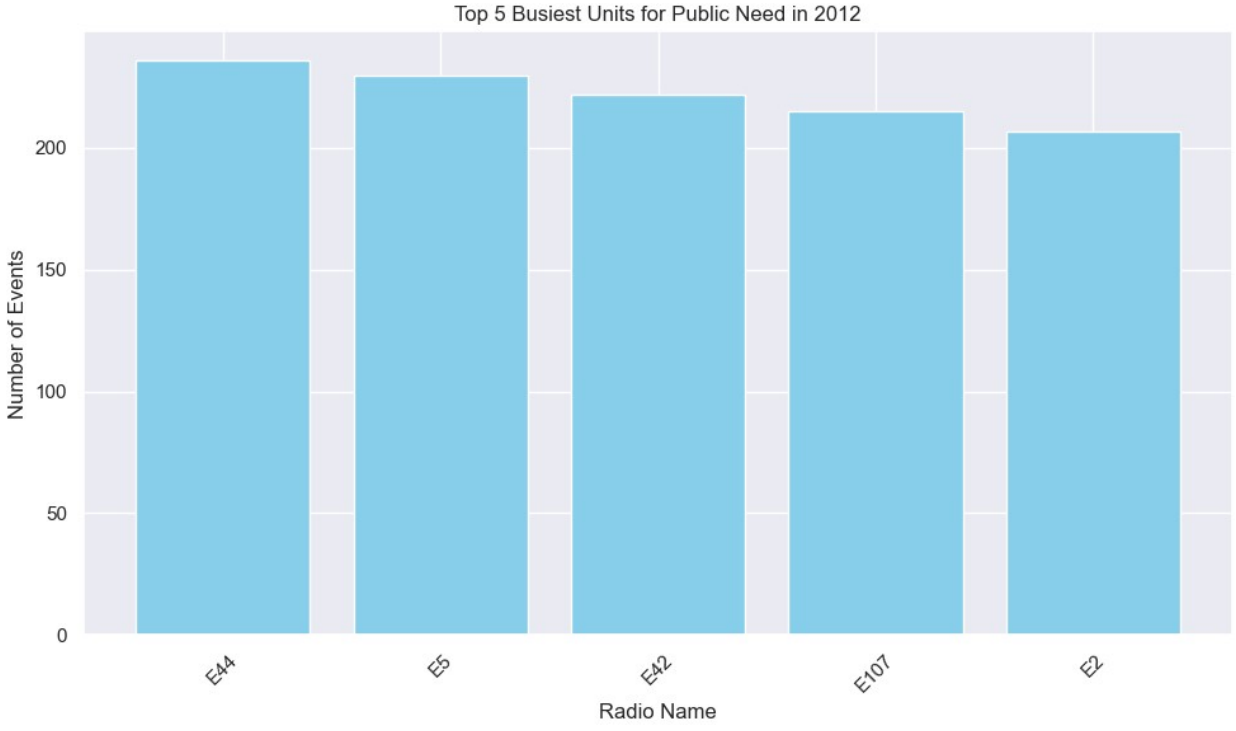
```python
# Group by year, event type, and radio name, then count the number of events
events_by_unit = df.groupby(['Year_Extracted', 'Event_Type', 'Radio_Name']).size().reset_index(name='Events')

# Function to plot the top 5 busiest units for each event type and year
def plot_top_5_units(year):
    # Filter data for the given year
    df_year = events_by_unit[events_by_unit['Year_Extracted'] == year]

    # Get unique event types for the year
    event_types = df_year['Event_Type'].unique()

    # Plot for each event type
    for event_type in event_types:
        # Filter data for the event type
        df_event_type = df_year[df_year['Event_Type'] == event_type]

        # Sort by events in descending order
        df_event_type = df_event_type.sort_values(by='Events', ascending=False)

        # Take top 5 busiest units
        top_5_units = df_event_type.head(5)

        # Plot
        plt.figure(figsize=(10, 6))
        plt.bar(top_5_units['Radio_Name'], top_5_units['Events'], color='skyblue')
        plt.xlabel('Radio Name')
        plt.ylabel('Number of Events')
        plt.title(f'Top 5 Busiest Units for {event_type} in {year}')
        plt.xticks(rotation=45)
        plt.tight_layout()
```

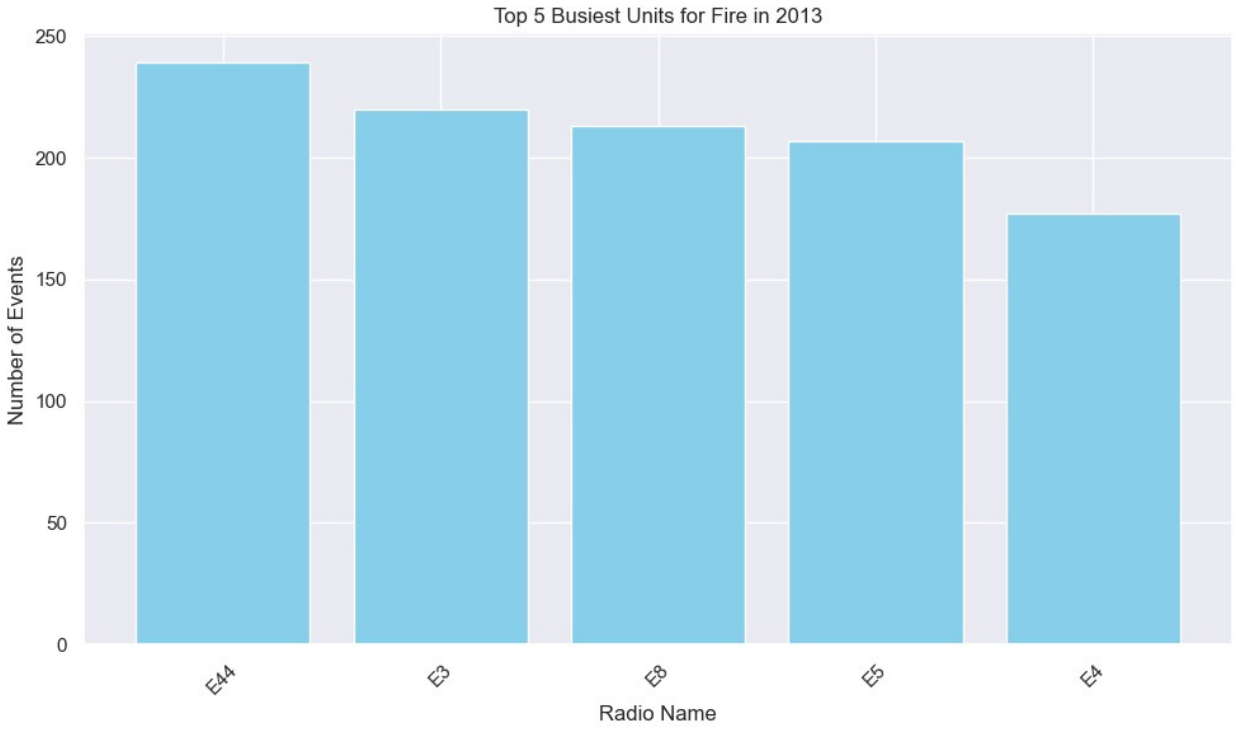
```
        plt.show()

# Plot for each year
for year in df['Year_Extracted'].unique():
    plot_top_5_units(year)
```

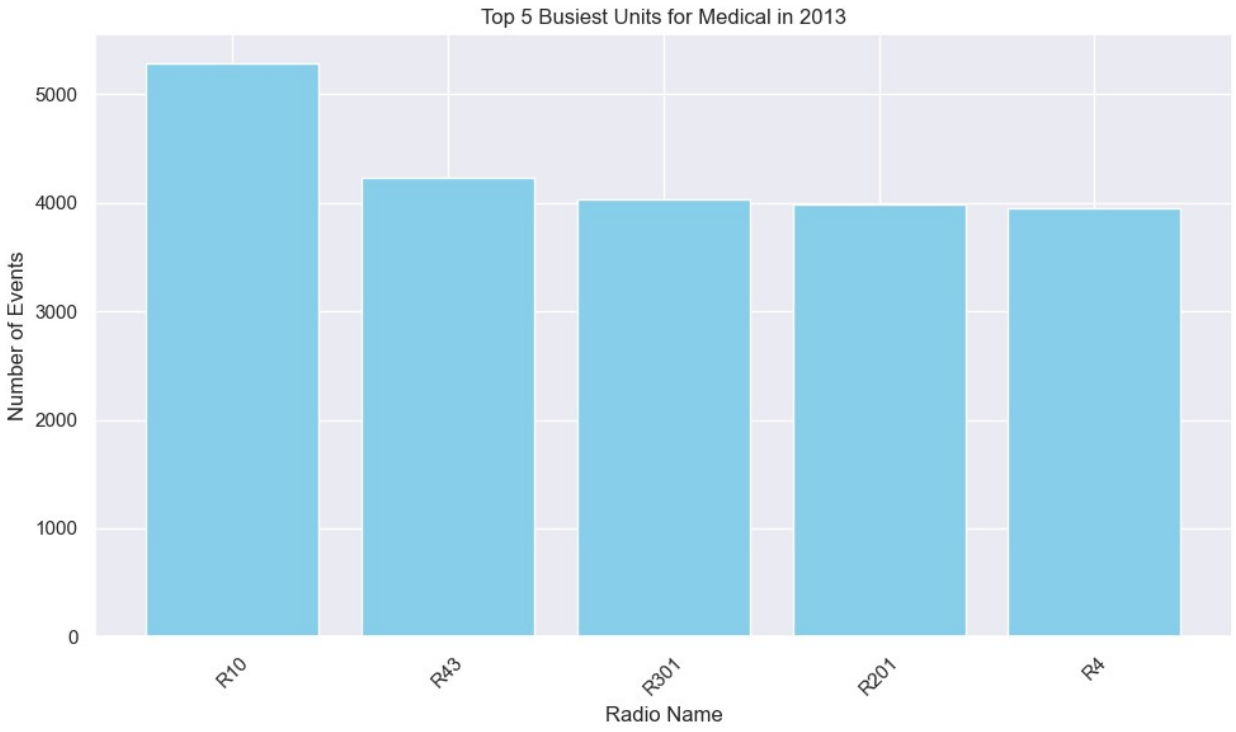


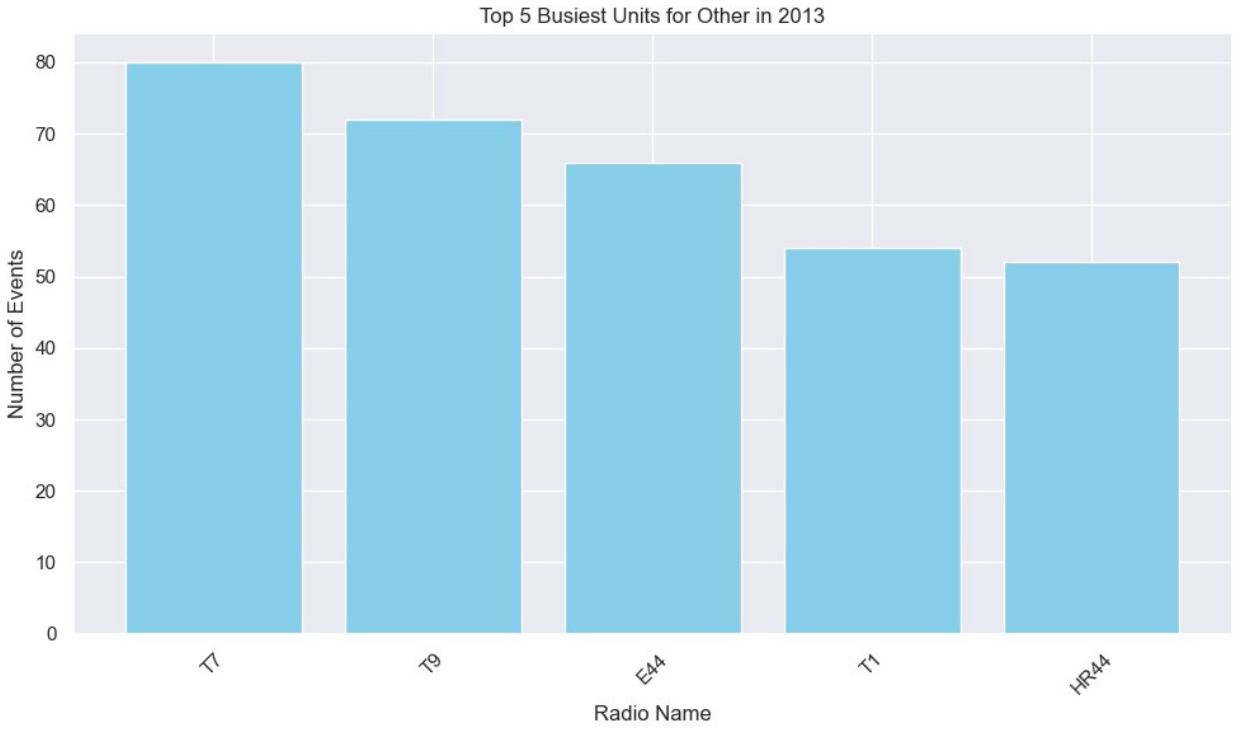


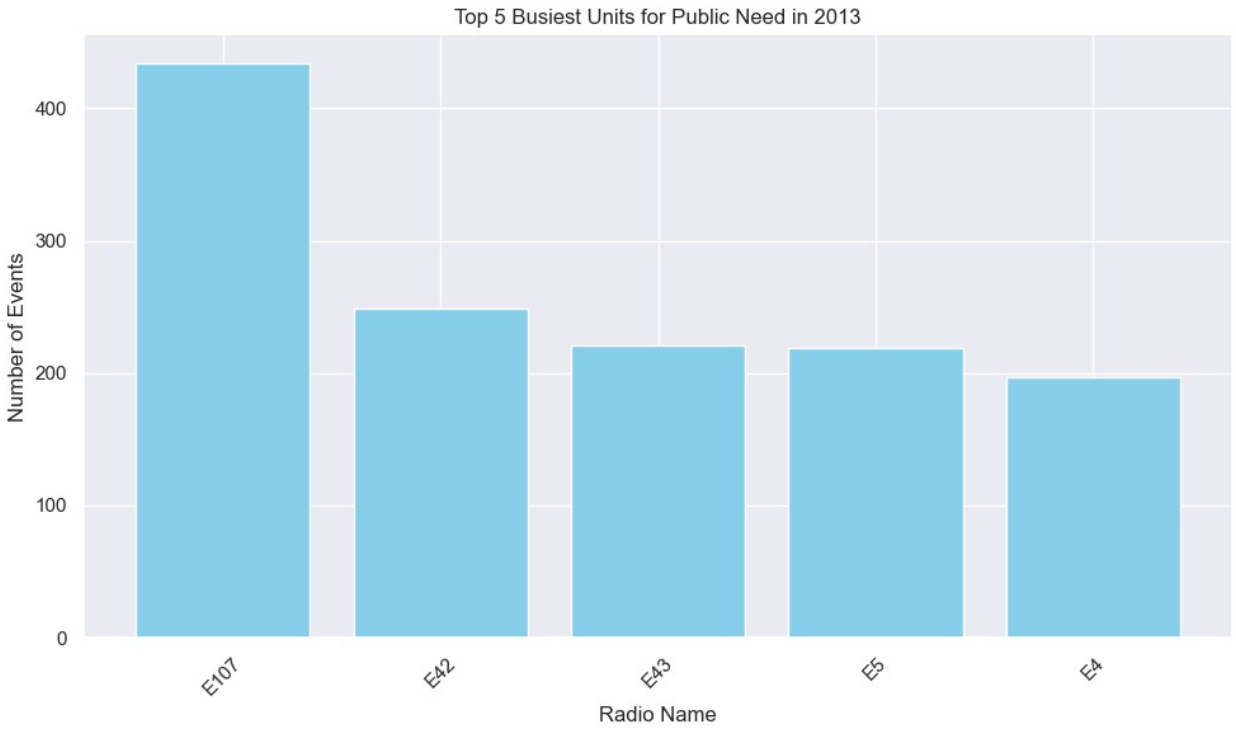

Top 5 Busiest Units for Other in 2012
Number of Events
0
10
20
30
40
50
60
70
80
T9
T7
E44
E8
T1
Radio Name

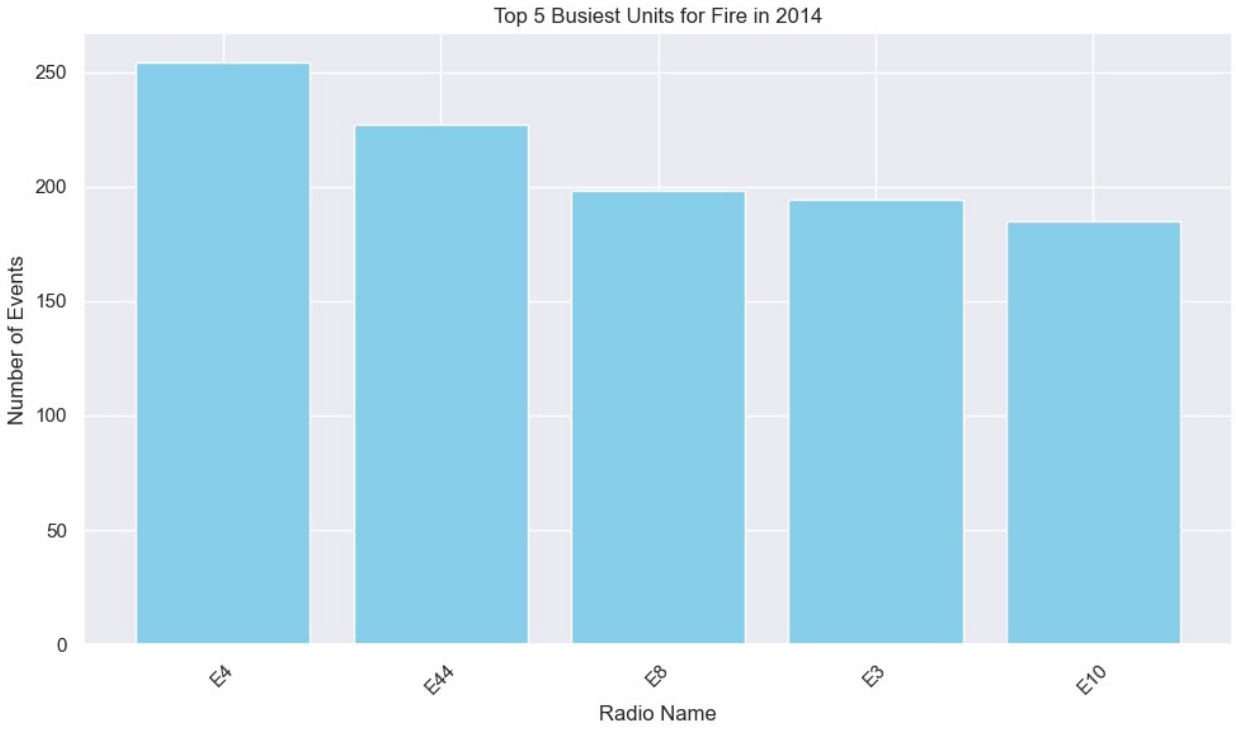

Top 5 Busiest Units for Public Need in 2012
Number of Events
0
50
100
150
200
E44
E5
E42
E107
E2
Radio Name

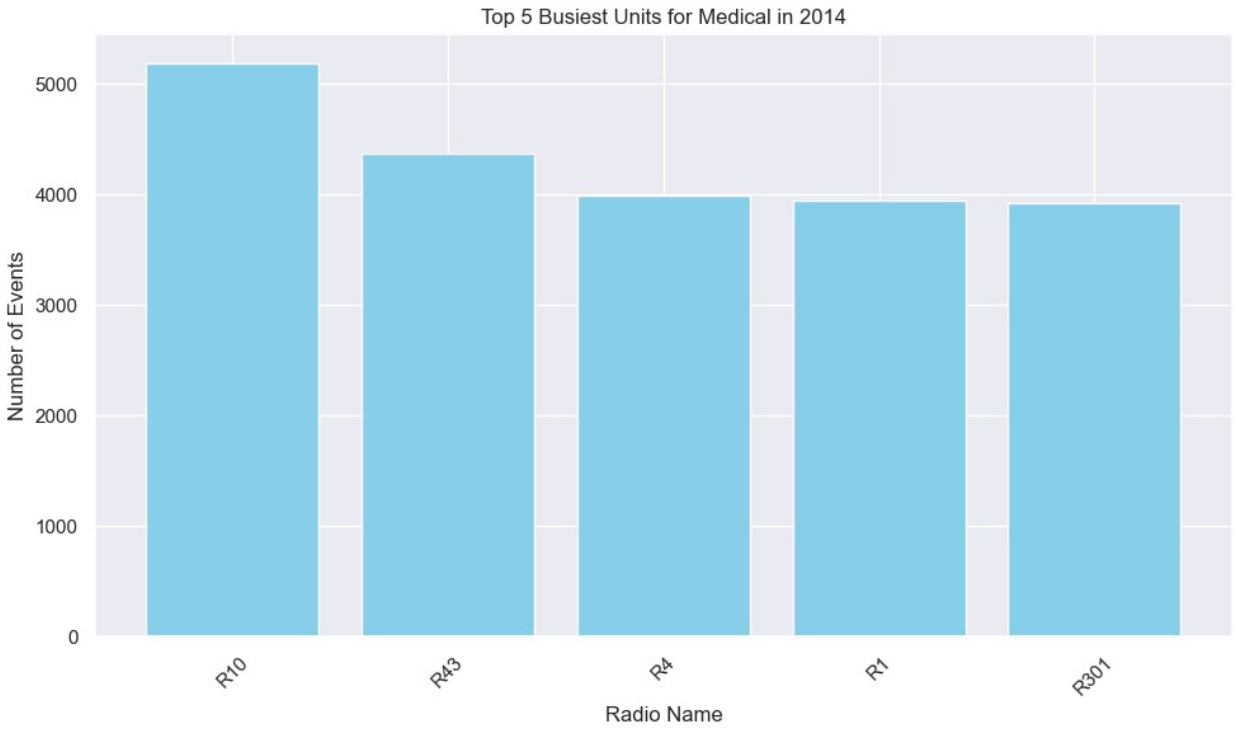

Top 5 Busiest Units for Fire in 2013
Number of Events
250
200
150
100
50
0
E44
E3
E8
E5
E4
Radio Name

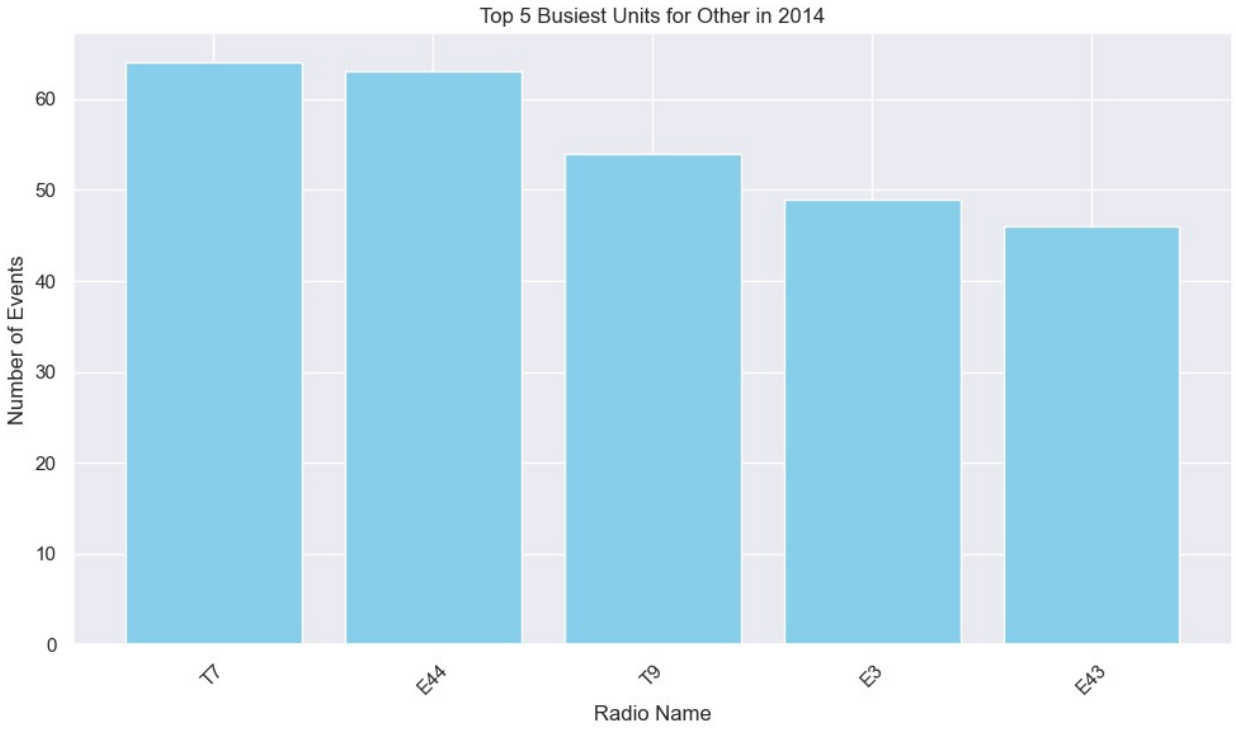

Top 5 Busiest Units for Medical in 2013
Number of Events
5000
4000
3000
2000
1000
0
R10
R43
R301
R201
R4
Radio Name

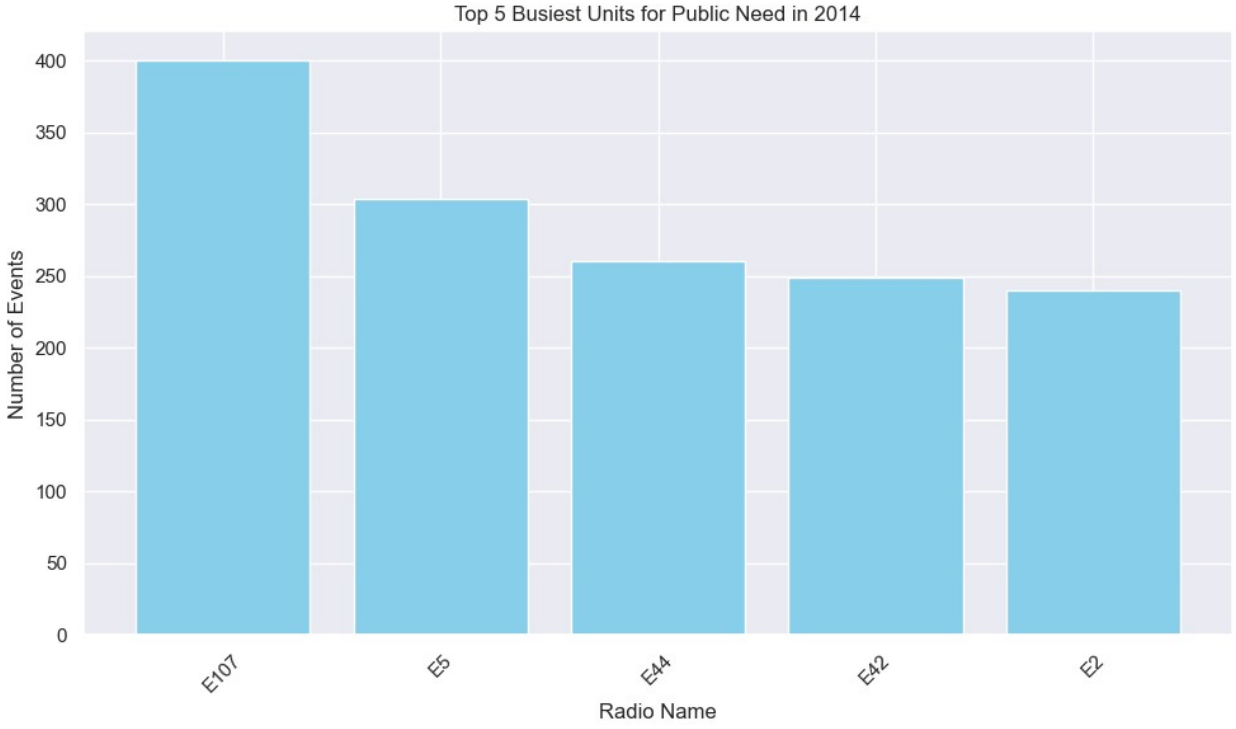

Top 5 Busiest Units for Other in 2013
Number of Events
0
10
20
30
40
50
60
70
80
T7
T9
E44
T1
HR44
Radio Name

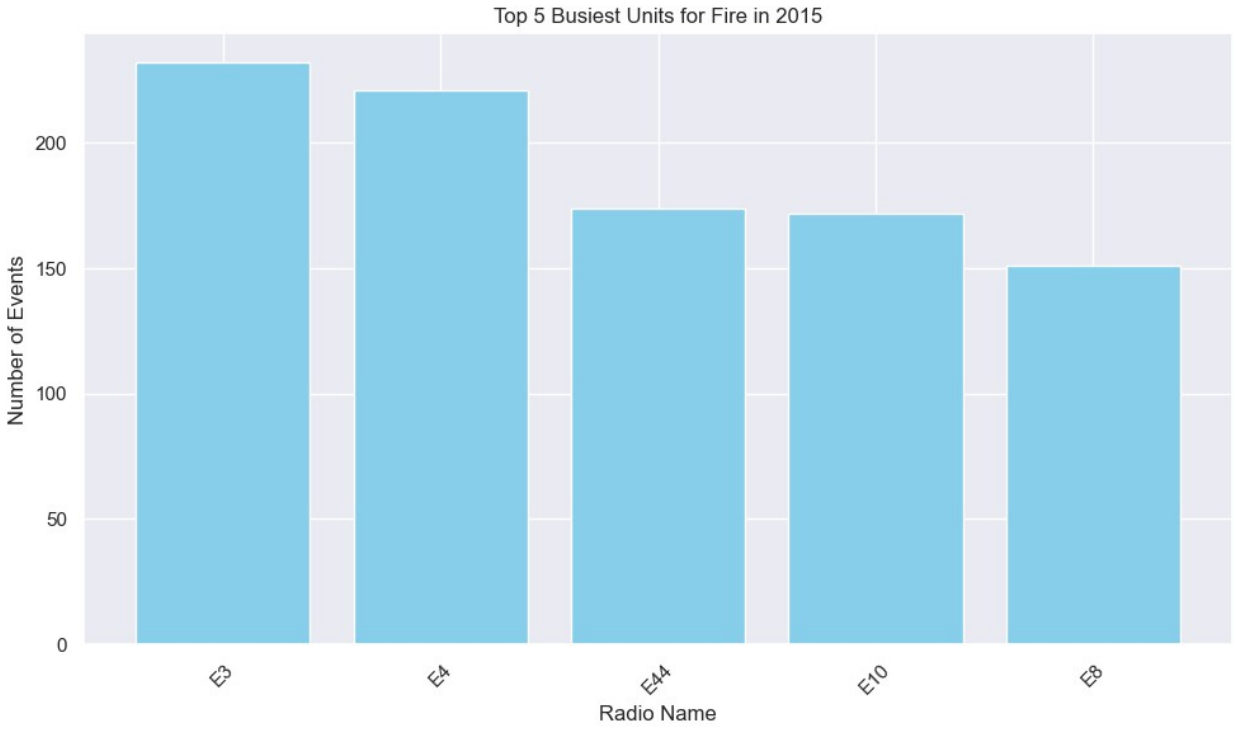

Top 5 Busiest Units for Public Need in 2013
Number of Events
0
100
200
300
400
E107
E42
E43
E5
E4
Radio Name

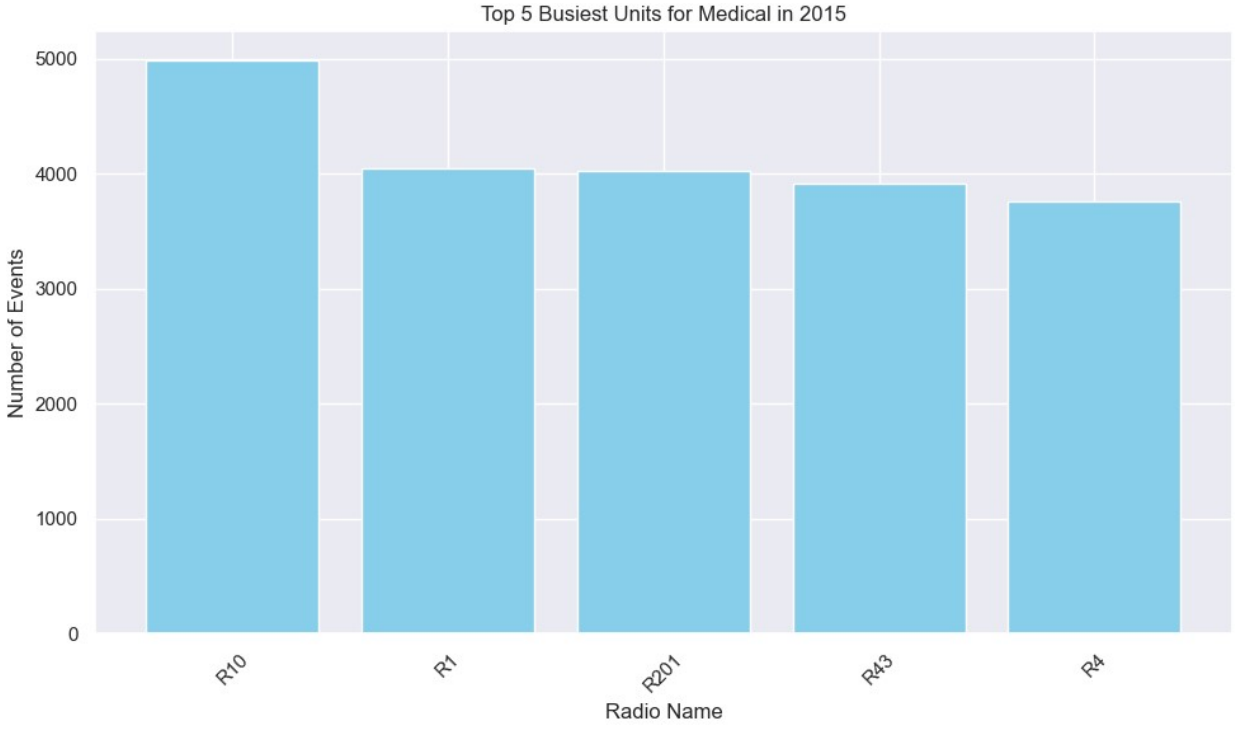

Top 5 Busiest Units for Fire in 2014
Number of Events
0
50
100
150
200
250
E4
E44
E8
E3
E10
Radio Name

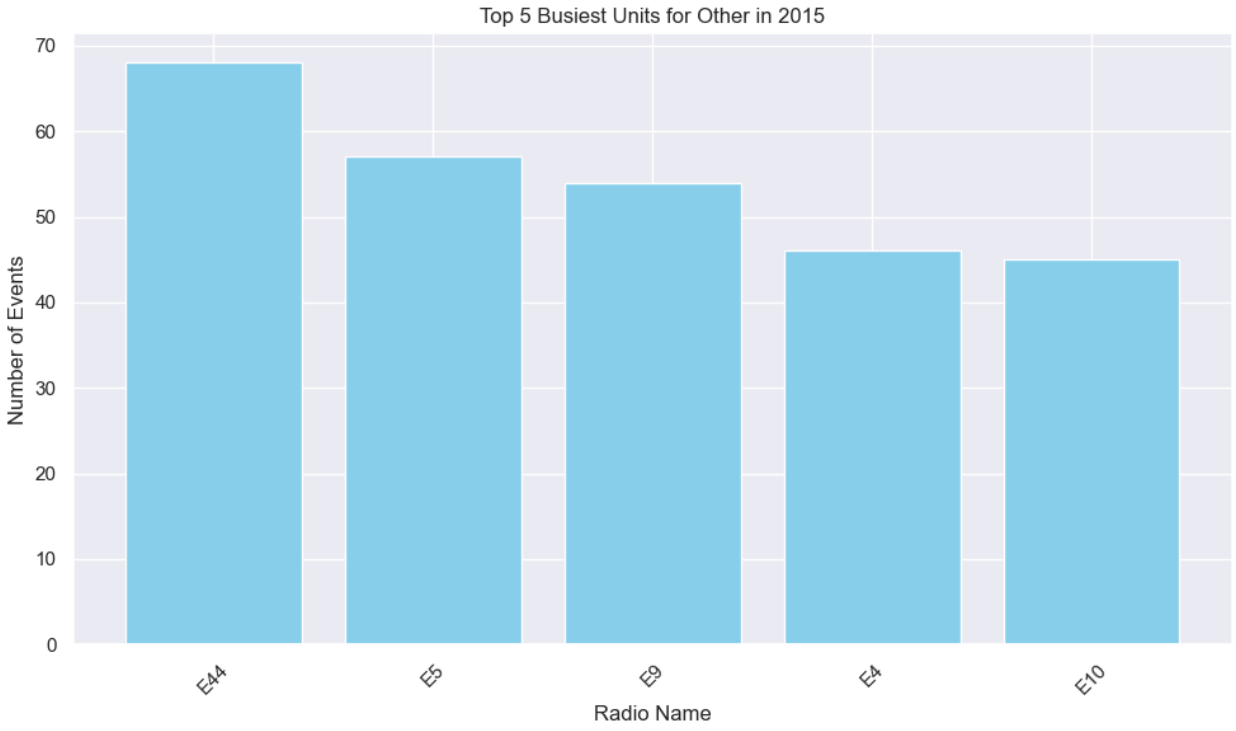

Top 5 Busiest Units for Medical in 2014
Number of Events
0
1000
2000
3000
4000
5000
R10
R43
R4
R1
R301
Radio Name

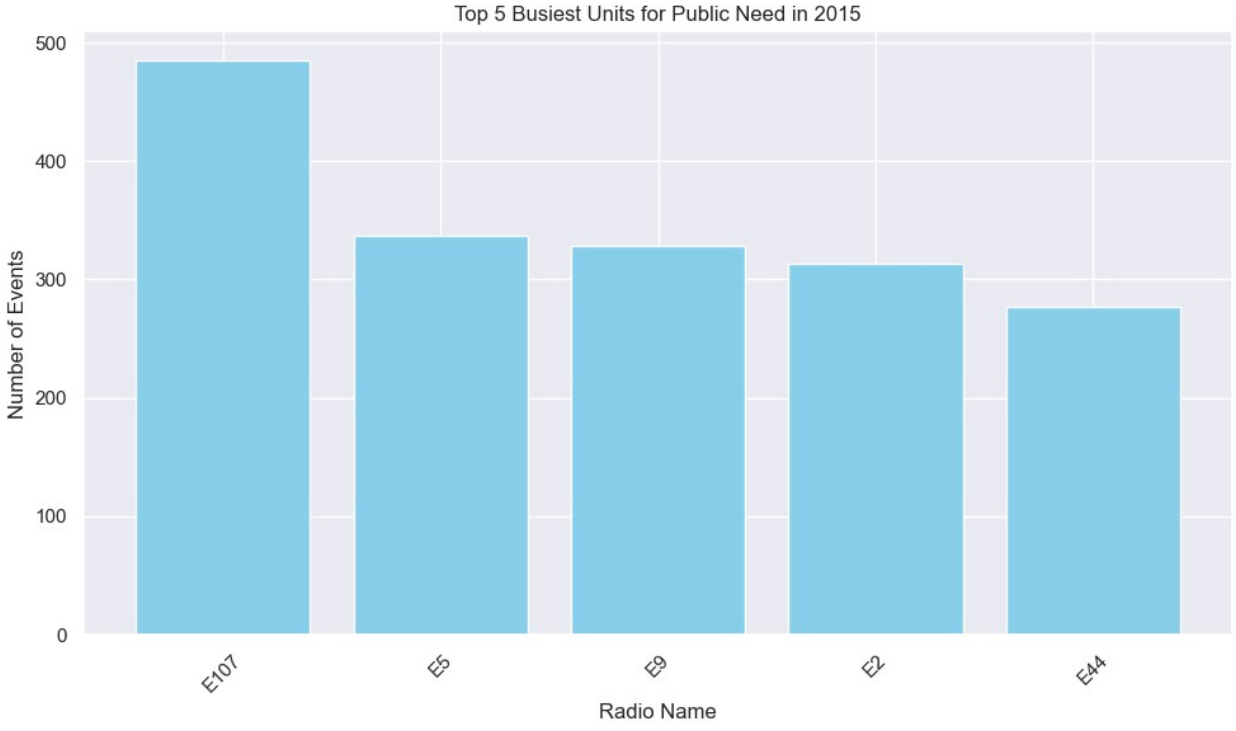

Top 5 Busiest Units for Other in 2014
Number of Events
0
10
20
30
40
50
60
T7
E44
T9
E3
E43
Radio Name

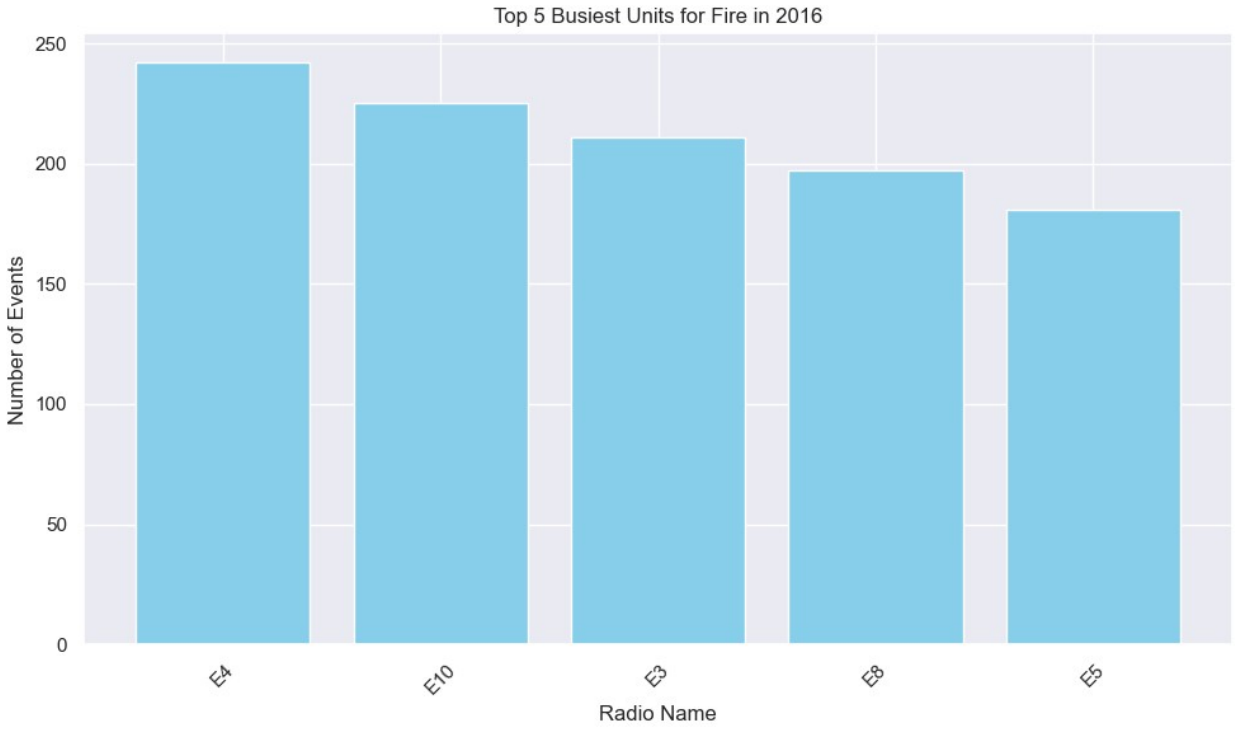

Top 5 Busiest Units for Public Need in 2014
Number of Events
0
50
100
150
200
250
300
350
400
E107
E5
E44
E42
E2
Radio Name

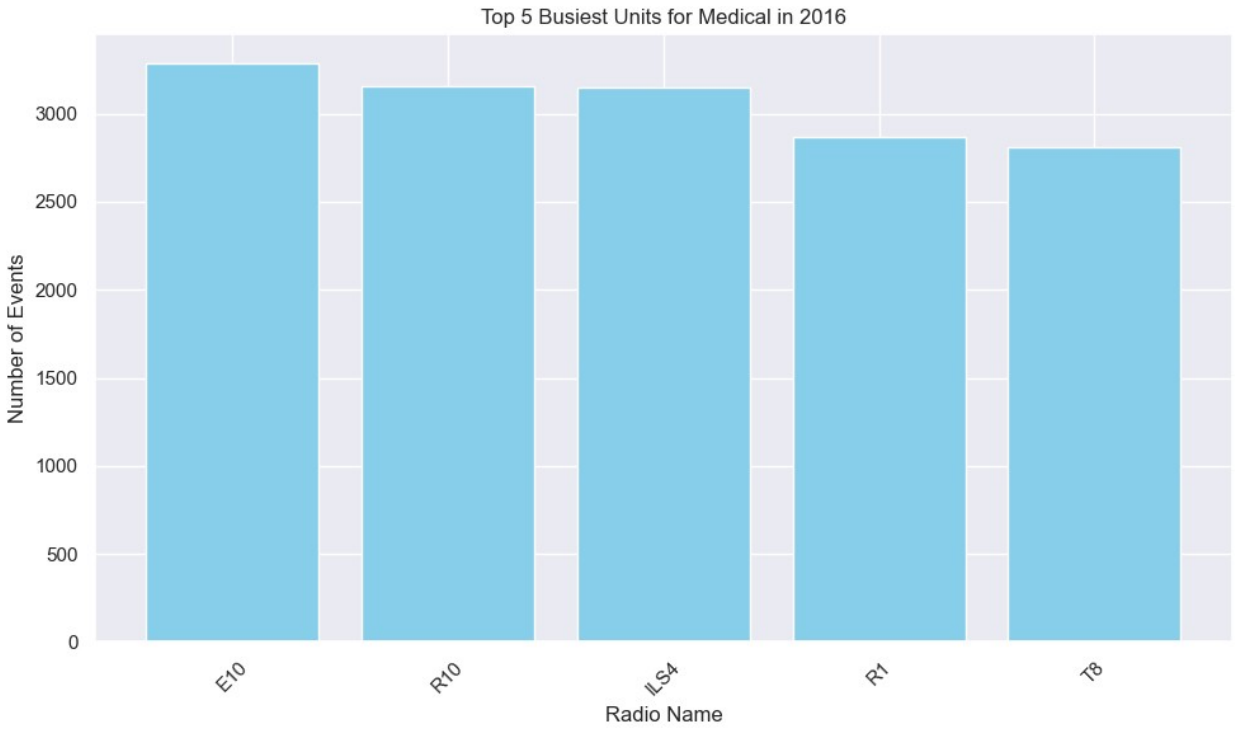

Top 5 Busiest Units for Fire in 2015
Number of Events
0
50
100
150
200
E3
E4
E44
E10
E8
Radio Name

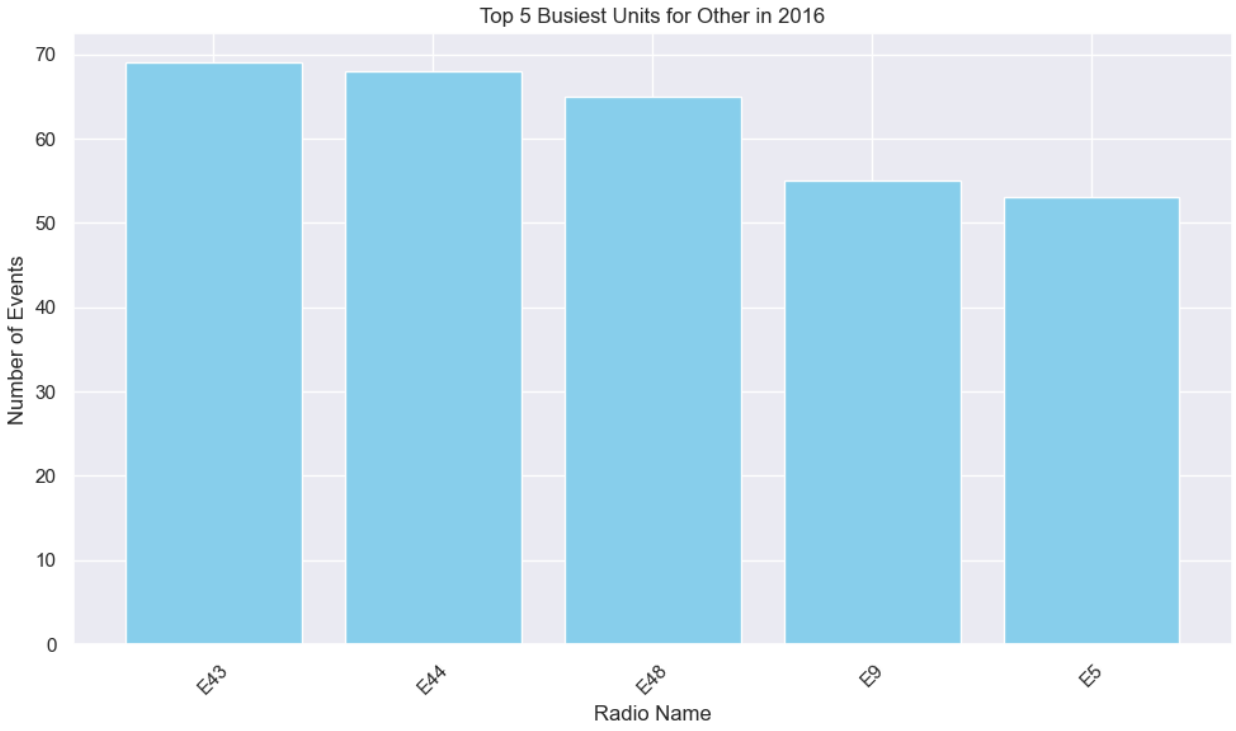

Top 5 Busiest Units for Medical in 2015
Number of Events
0
1000
2000
3000
4000
5000
R10
R1
R201
R43
R4
Radio Name

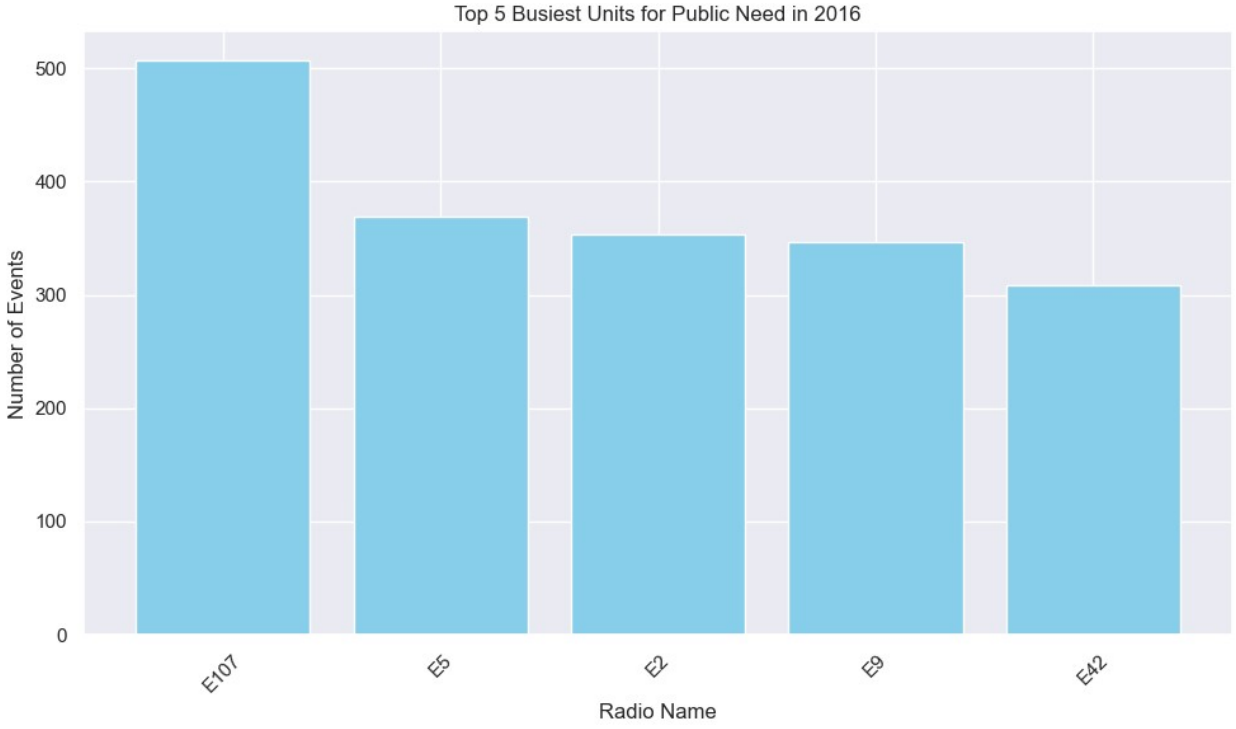

Top 5 Busiest Units for Other in 2015
Number of Events
0
10
20
30
40
50
60
70
E44
E5
E9
E4
E10
Radio Name

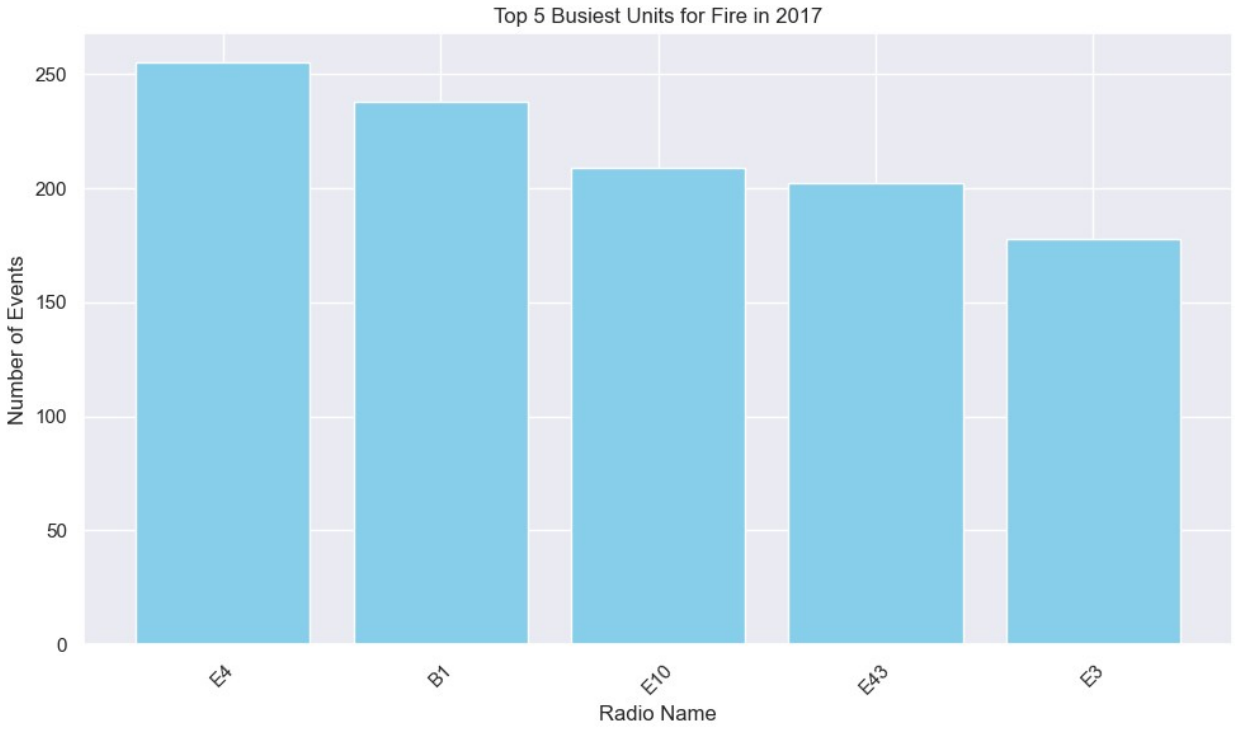

Top 5 Busiest Units for Public Need in 2015
Number of Events
0
100
200
300
400
500
E107
E5
E9
E2
E44
Radio Name

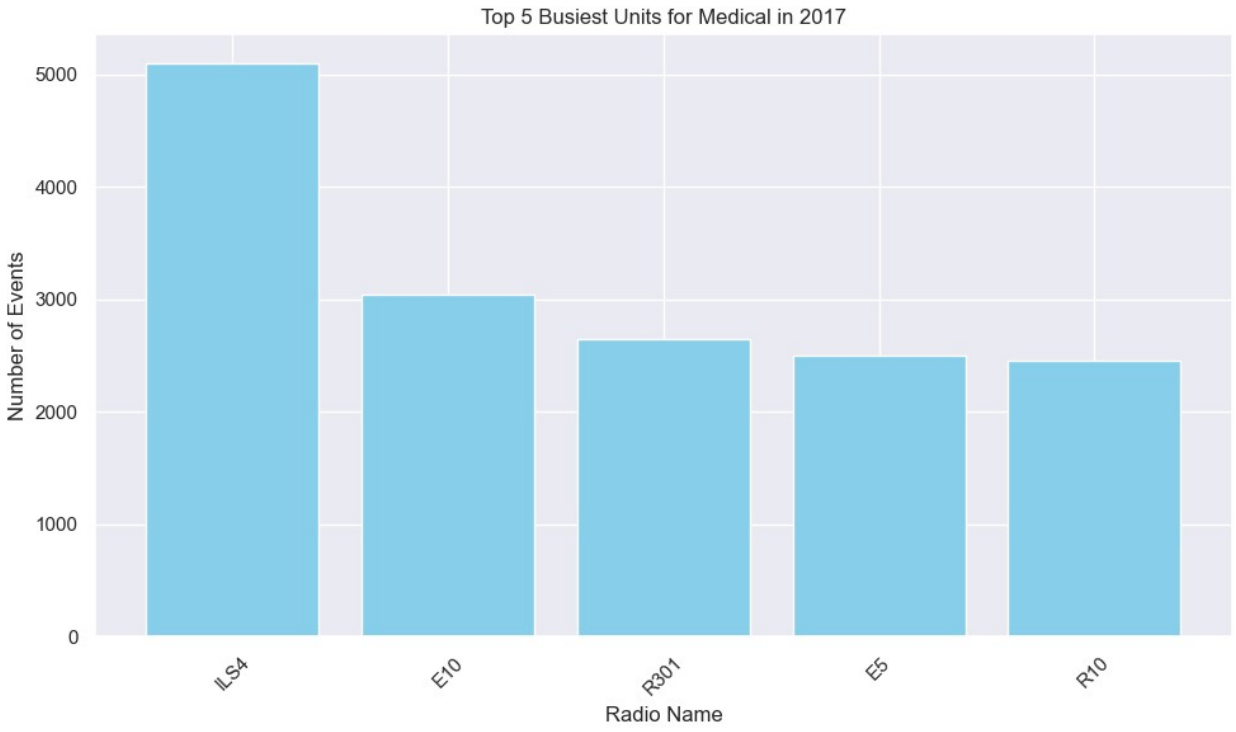

Top 5 Busiest Units for Fire in 2016
Number of Events
250
200
150
100
50
0
E4
E10
E3
E8
E5
Radio Name

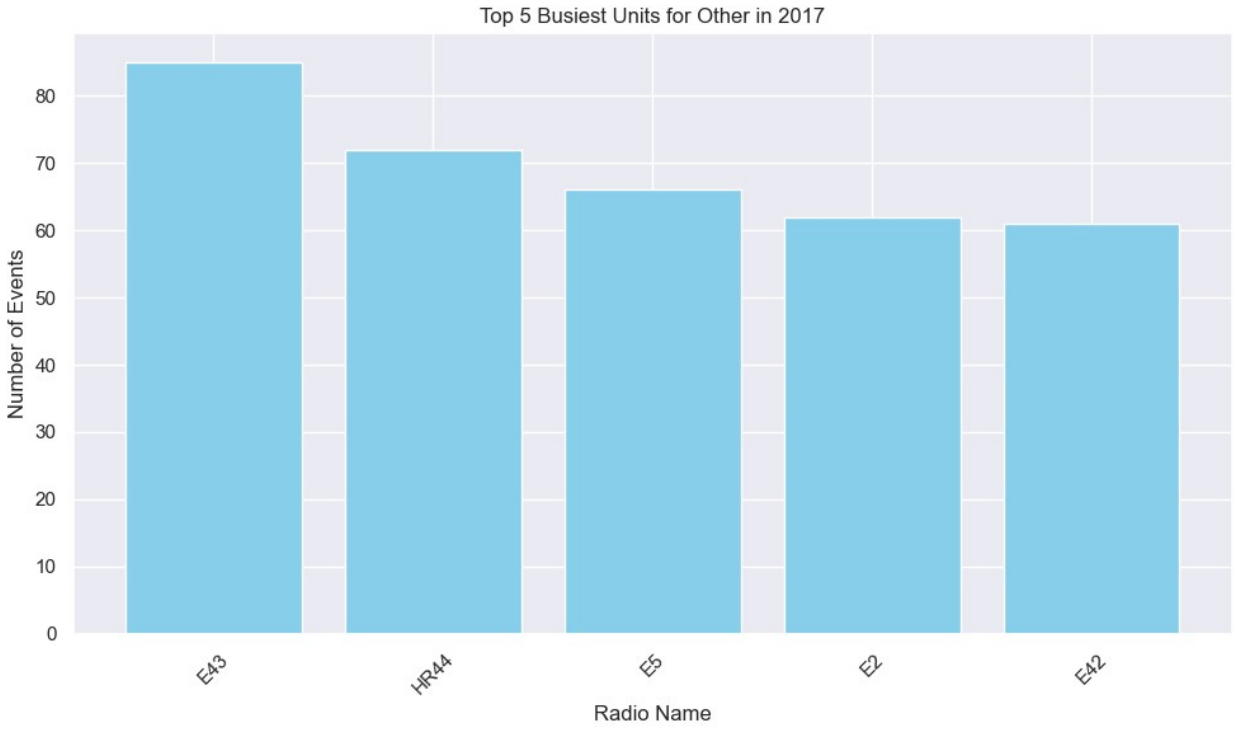

Top 5 Busiest Units for Medical in 2016
Number of Events
3000
2500
2000
1500
1000
500
0
E10
R10
ILS4
R1
T8
Radio Name

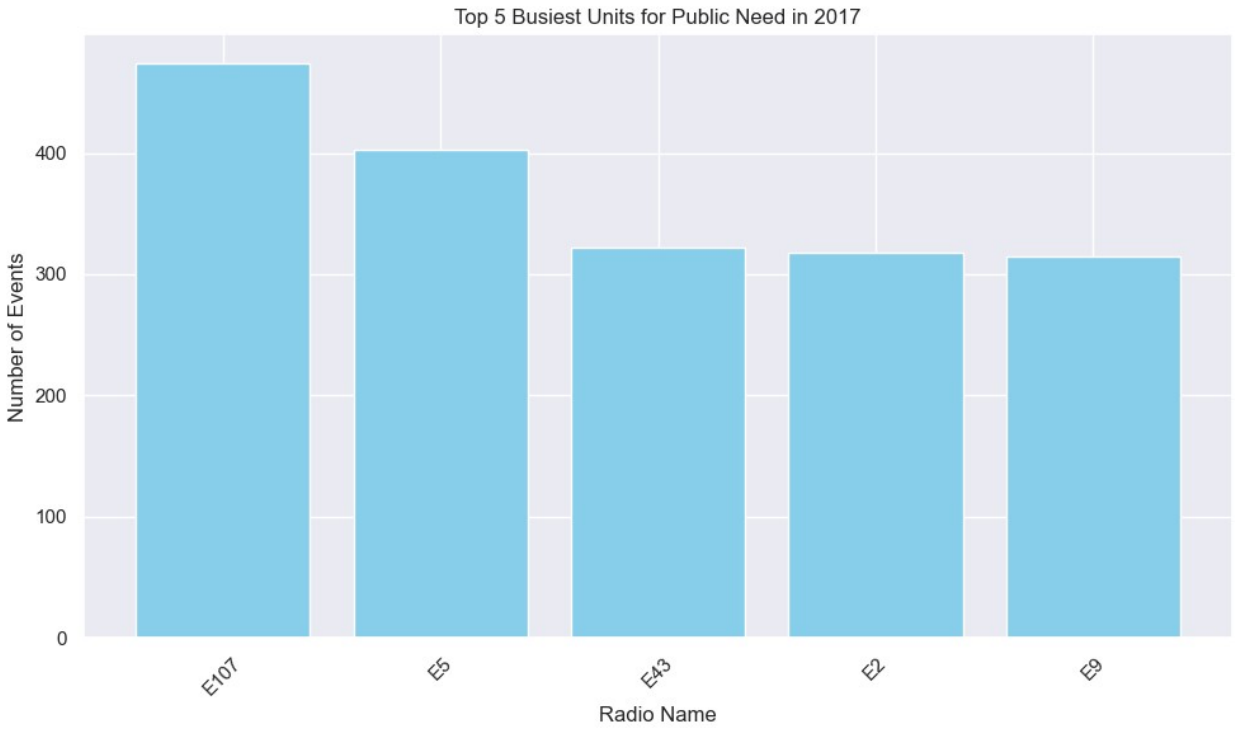

Top 5 Busiest Units for Other in 2016
Number of Events
0
10
20
30
40
50
60
70
E43
E44
E48
E9
E5
Radio Name

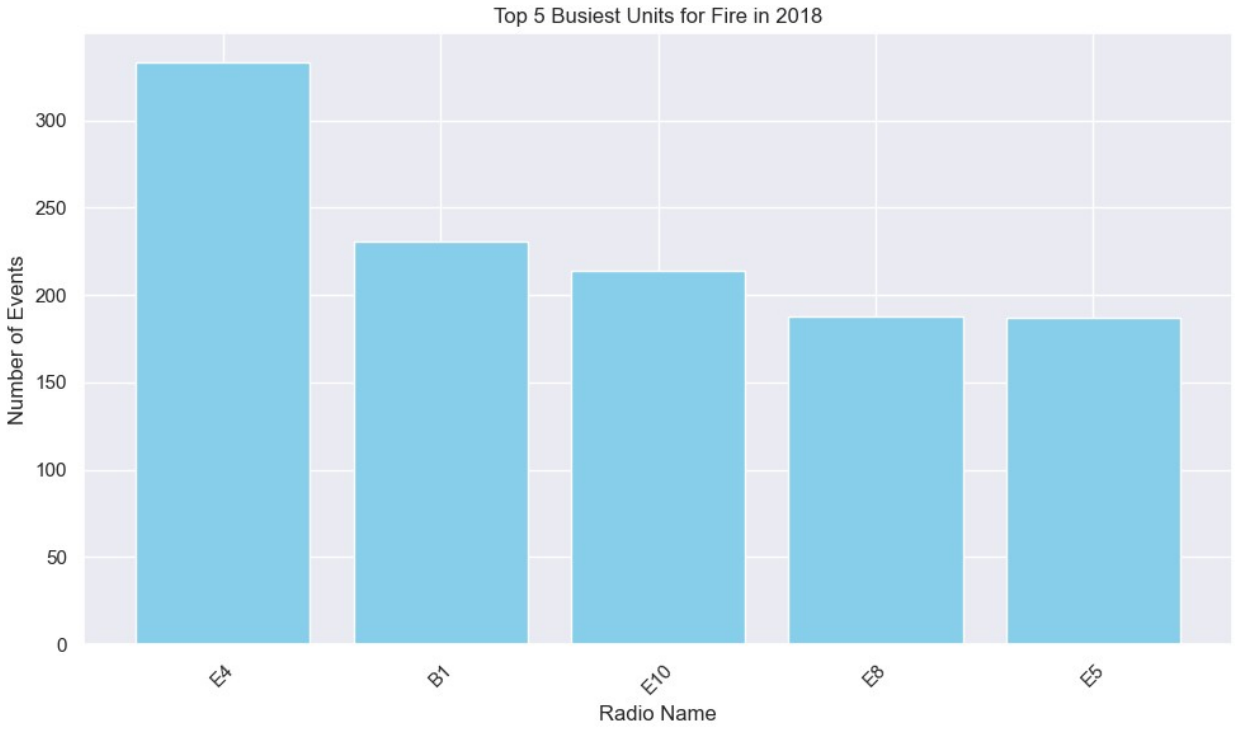

Top 5 Busiest Units for Public Need in 2016
Number of Events
0
100
200
300
400
500
E107
E5
E2
E9
E42
Radio Name

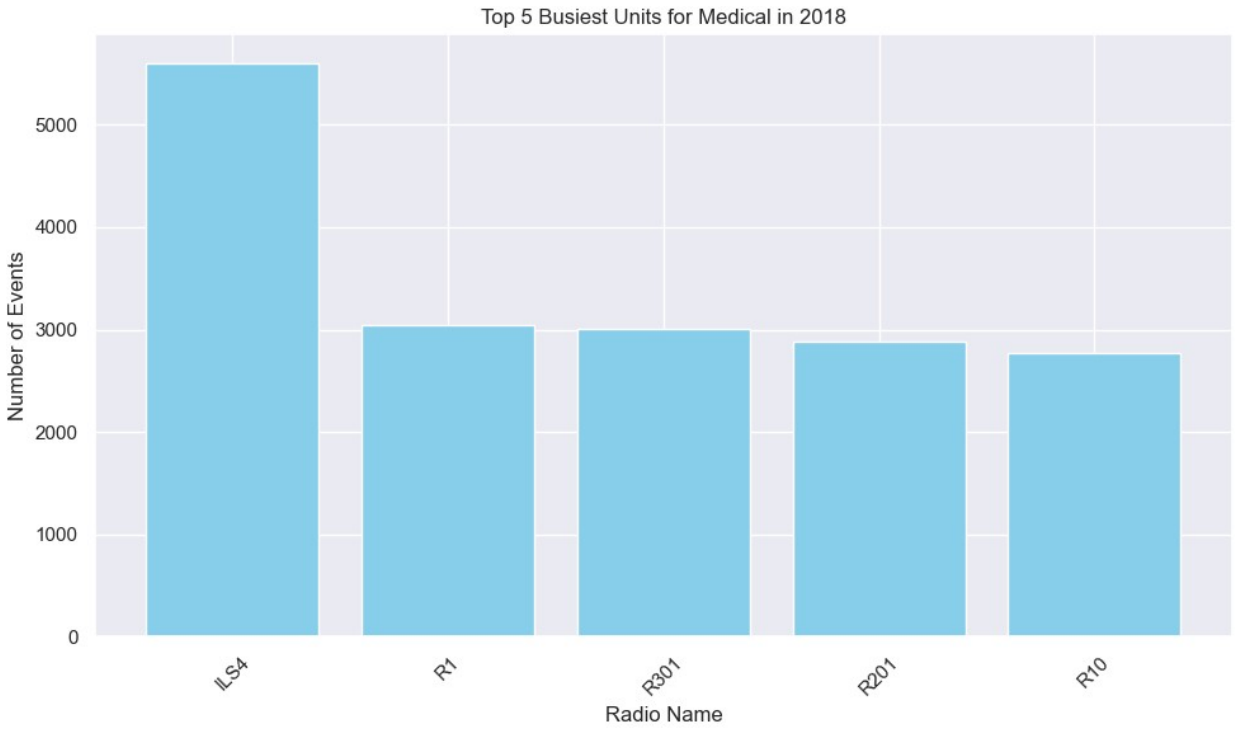

Top 5 Busiest Units for Fire in 2017
Number of Events
0
50
100
150
200
250
E4
B1
E10
E43
E3
Radio Name

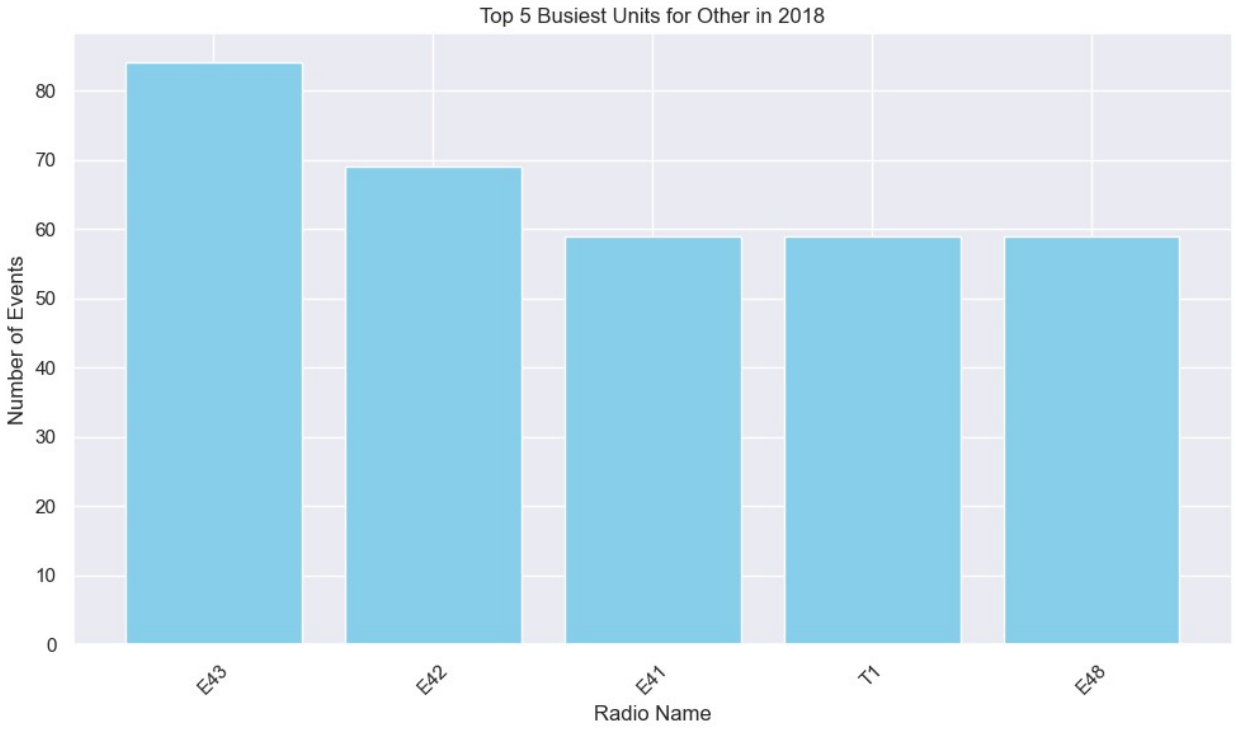

Top 5 Busiest Units for Medical in 2017
Number of Events
0
1000
2000
3000
4000
5000
ILS4
E10
R301
E5
R10
Radio Name

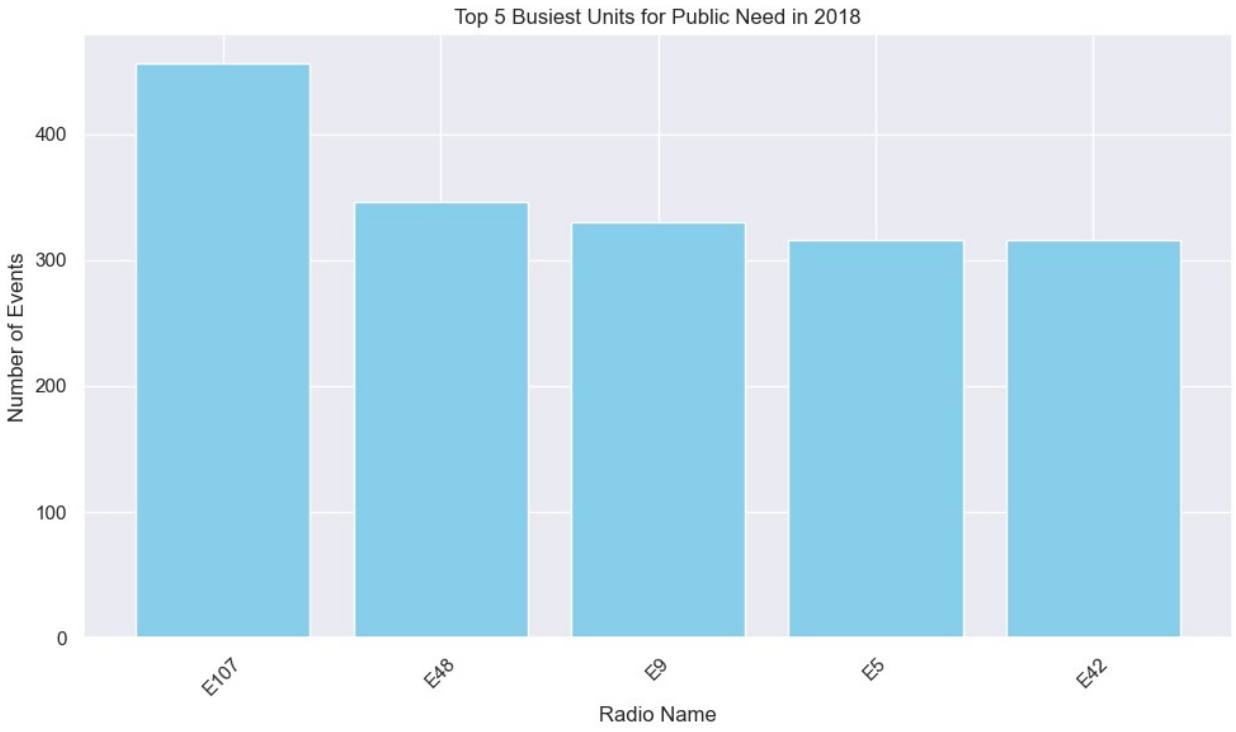

Top 5 Busiest Units for Other in 2017
Number of Events
0
10
20
30
40
50
60
70
80
E43
HR44
E5
E2
E42
Radio Name


Top 5 Busiest Units for Public Need in 2017
Number of Events
0
100
200
300
400
E107
E5
E43
E2
E9
Radio Name


Top 5 Busiest Units for Fire in 2018
Number of Events
0
50
100
150
200
250
300
E4
B1
E10
E8
E5
Radio Name

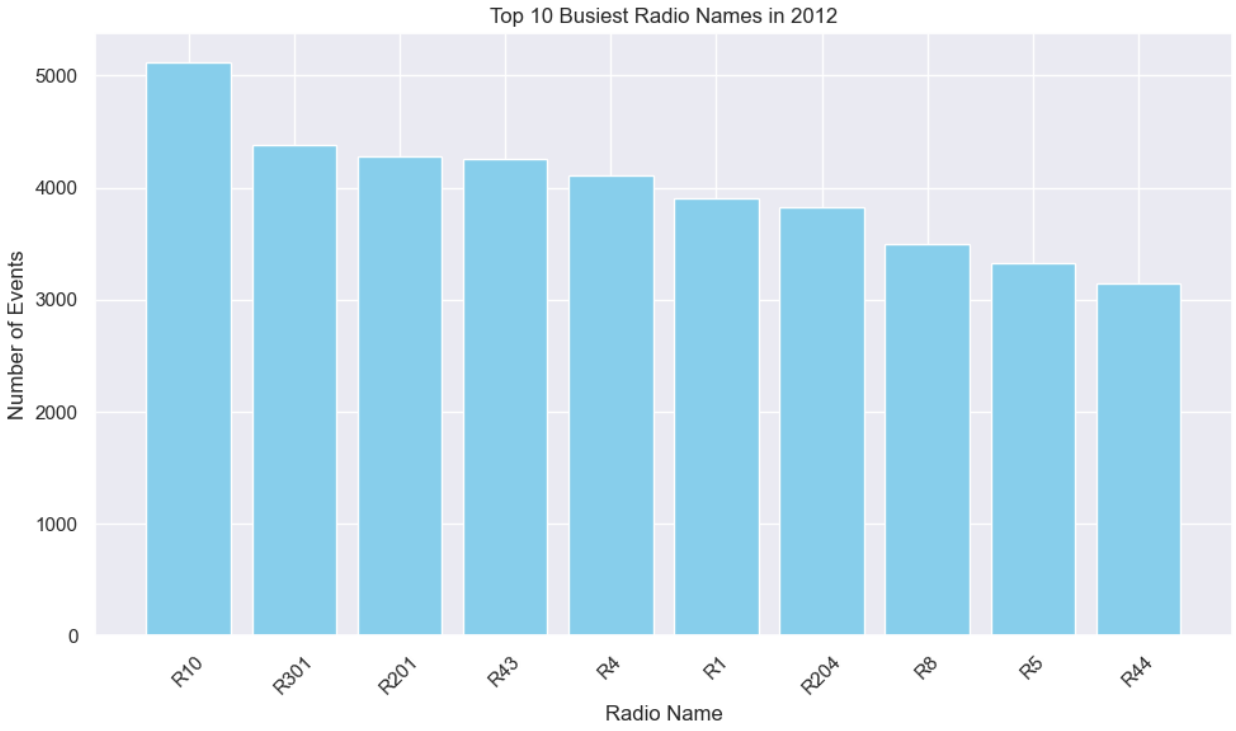

Top 5 Busiest Units for Medical in 2018
Number of Events
0
1000
2000
3000
4000
5000
ILS4
R1
R301
R201
R10
Radio Name

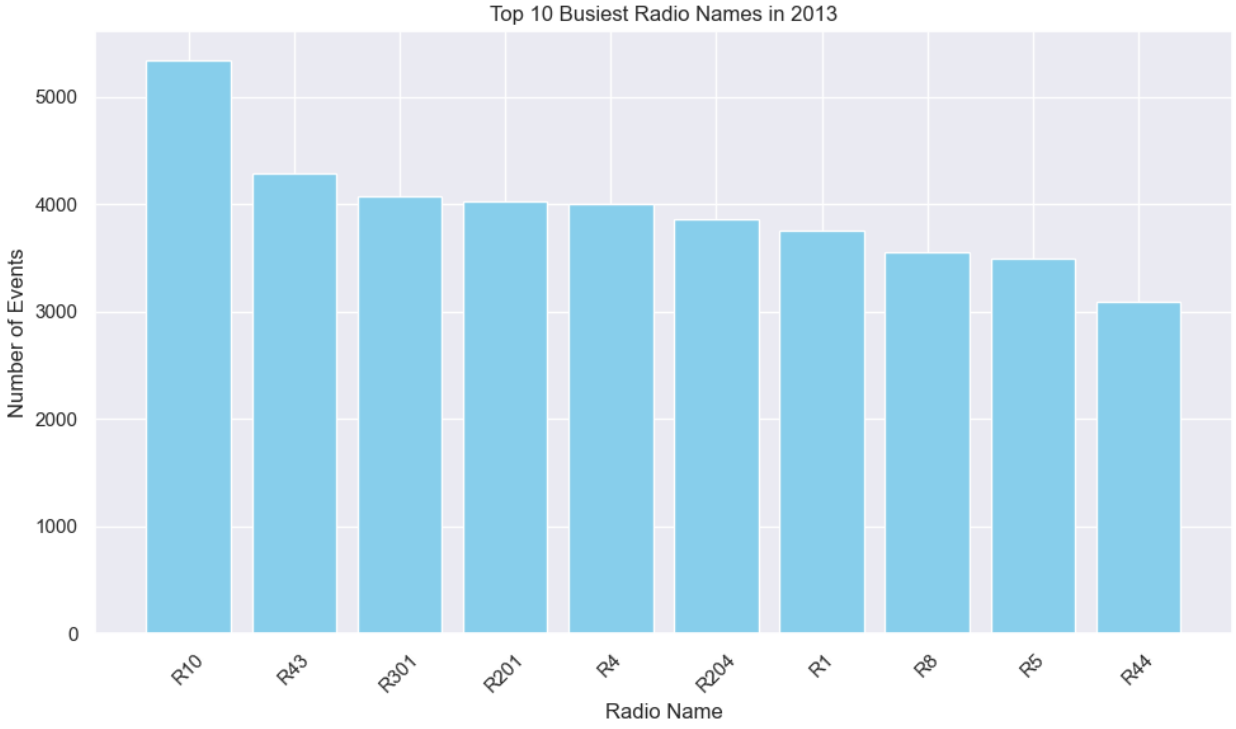


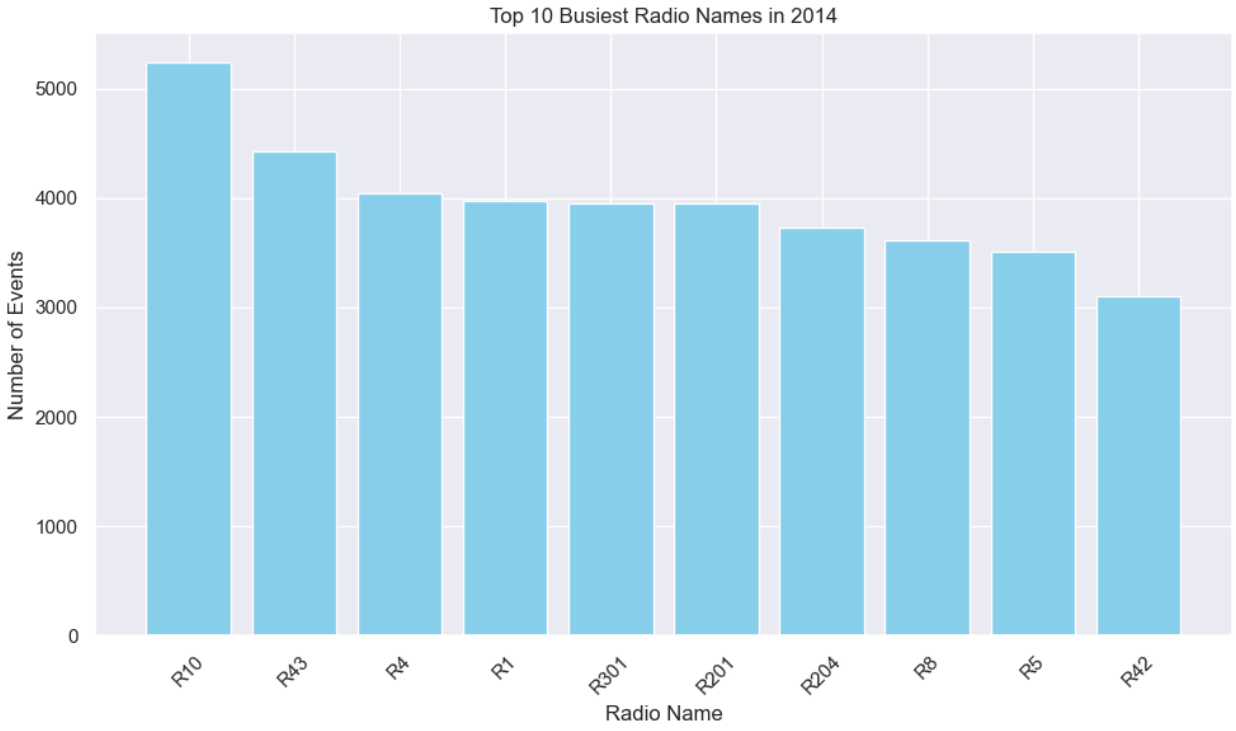


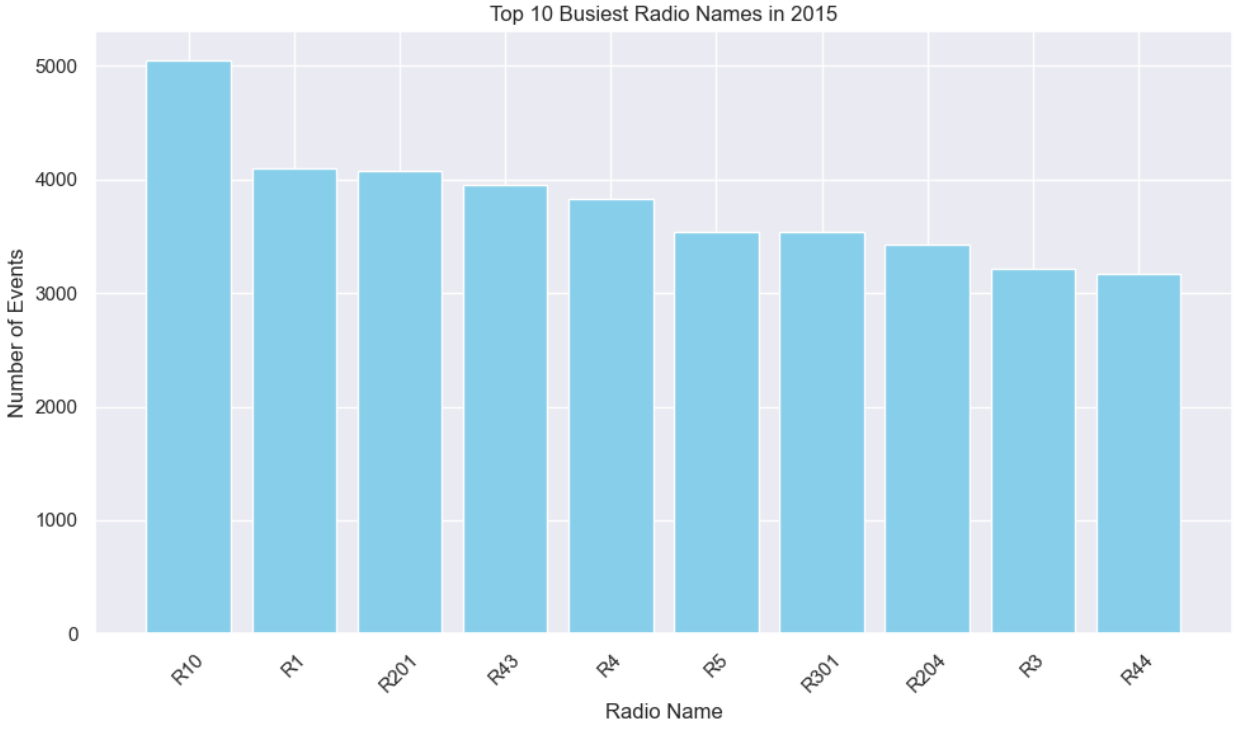

```
# Group by year and radio name, then count the number of events
events_by_radio = df.groupby(['Year_Extracted',
'Radio_Name']).size().reset_index(name='Events')

# Function to plot the top 10 radio names for each year
```

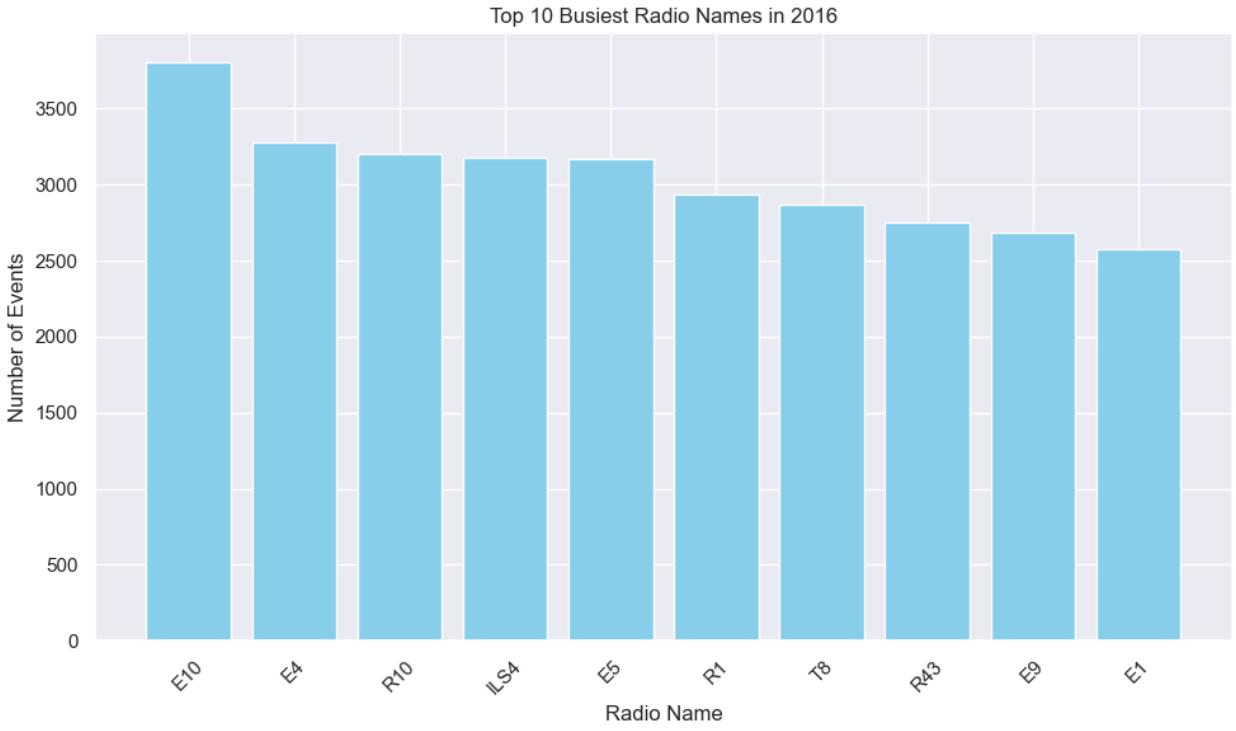
```
def plot_top_10_radio(year):
    # Filter data for the given year
    df_year = events_by_radio[events_by_radio['Year_Extracted'] ==
year]

    # Sort by events in descending order
    df_year = df_year.sort_values(by='Events', ascending=False)

    # Take top 10 busiest radio names
    top_10_radio = df_year.head(10)

    # Plot
    plt.figure(figsize=(10, 6))
    plt.bar(top_10_radio['Radio_Name'], top_10_radio['Events'],
color='skyblue')
    plt.xlabel('Radio Name')
    plt.ylabel('Number of Events')
    plt.title(f'Top 10 Busiest Radio Names in {year}')
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

# Plot for each year
for year in df['Year_Extracted'].unique():
    plot_top_10_radio(year)
```

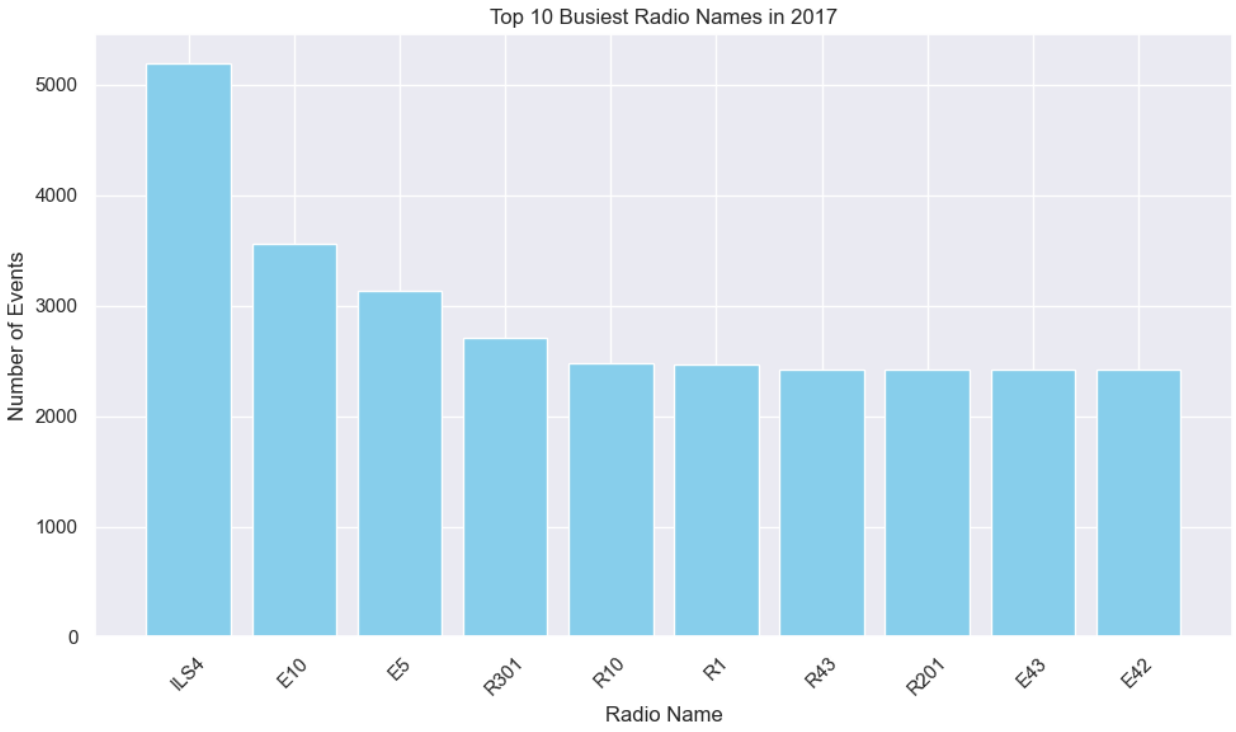

Top 10 Busiest Radio Names in 2012
Number of Events
0
1000
2000
3000
4000
5000
R10
R301
R201
R43
R4
R1
R204
R8
R5
R44
Radio Name

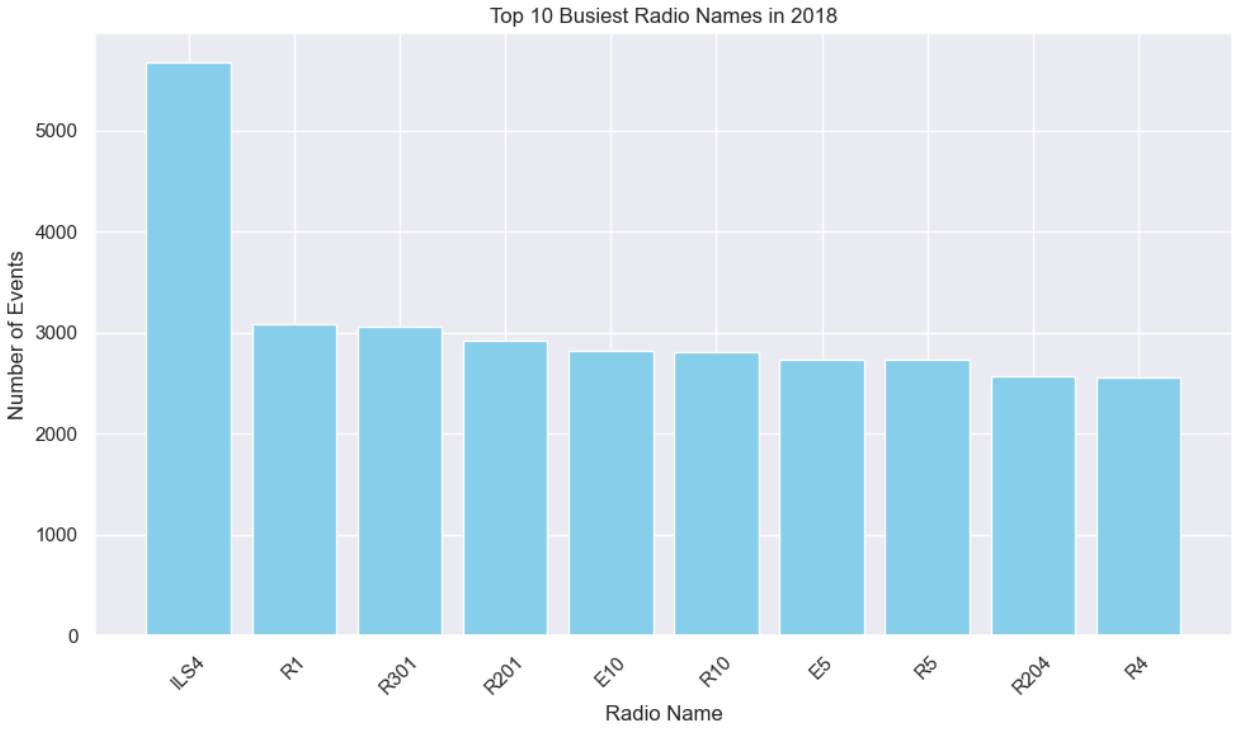

Top 10 Busiest Radio Names in 2013
Number of Events
0
1000
2000
3000
4000
5000
R10
R43
R301
R201
R4
R204
R1
R8
R5
R44
Radio Name


Top 10 Busiest Radio Names in 2014
Number of Events
0
1000
2000
3000
4000
5000
R10
R43
R4
R1
R301
R201
R204
R8
R5
R42
Radio Name


Top 10 Busiest Radio Names in 2015
Number of Events
0
1000
2000
3000
4000
5000
R10
R1
R201
R43
R4
R5
R301
R204
R3
R44
Radio Name


Top 10 Busiest Radio Names in 2016
Number of Events
0
500
1000
1500
2000
2500
3000
3500
E10
E4
R10
ILS4
E5
R1
T8
R43
E9
E1
Radio Name


Top 10 Busiest Radio Names in 2017
Number of Events
0
1000
2000
3000
4000
5000
ILS4
E10
E5
R301
R10
R1
R43
R201
E43
E42
Radio Name


Top 10 Busiest Radio Names in 2018
Number of Events
0
1000
2000
3000
4000
5000
ILS4
R1
R301
R201
E10
R10
E5
R5
R204
R4
Radio Name