# CT_Drug_Deaths_Analysis

October 15, 2020

Let's start by reading our cleaned dataset in and inspecting it

[1]:
```
import pandas as pd
import os
from urllib.request import urlopen
import json
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
os.chdir('C:\\Users\\mhous\\CT\\CT-Drug-Deaths')
df = pd.read_csv('drug_deaths_clean.csv', index_col = ['ID'])

df.head()
```

[1]:
```
                 Date    Year      DateType   Age     Sex          Race  \
ID
14-0273  2014-06-28  2014.0  DateReported   NaN     NaN           NaN
13-0102  2013-03-21  2013.0   DateofDeath  48.0    Male         Black
16-0165  2016-03-13  2016.0   DateofDeath  30.0  Female         White
16-0208  2016-03-31  2016.0   DateofDeath  23.0    Male         White
13-0052  2013-02-13  2013.0   DateofDeath  22.0    Male  Asian, Other

        ResidenceCity ResidenceCounty ResidenceState  DeathCity  …  \
ID                                                               …
14-0273           NaN             NaN            NaN        NaN  …
13-0102       Norwalk             NaN            NaN    Norwalk  …
16-0165    Sandy Hook       Fairfield             CT    Danbury  …
16-0208           Rye     Westchester             NY  Greenwich  …
13-0052      Flushing          Queens            NaN  Greenwich  …

        Hydromorphone Other OpiateNOS AnyOpioid Medication Number_of_drugs  \
ID
14-0273             0     0         0         0          0               3
13-0102             0     0         0         0          0               1
16-0165             0     0         0         1          0               3
16-0208             0     0         0         1          0               3
13-0052             0     0         0         0          0               1
```

```
        MannerofDeath                            DeathCityGeo  \
ID
14-0273      Accident             CT\n(41.575155, -72.738288)
13-0102      Accident     Norwalk, CT\n(41.11805, -73.412906)
16-0165      Accident    Danbury, CT\n(41.393666, -73.451539)
16-0208      Accident  Greenwich, CT\n(41.026526, -73.628549)
13-0052      Accident  Greenwich, CT\n(41.026526, -73.628549)

                                ResidenceCityGeo              InjuryCityGeo
ID
14-0273              CT\n(41.575155, -72.738288)  CT\n(41.575155, -72.738288)
13-0102      NORWALK, CT\n(41.11805, -73.412906)  CT\n(41.575155, -72.738288)
16-0165  SANDY HOOK, CT\n(41.419998, -73.282501)                          NaN
16-0208                                      NaN                          NaN
13-0052                                      NaN  CT\n(41.575155, -72.738288)

[5 rows x 43 columns]
```

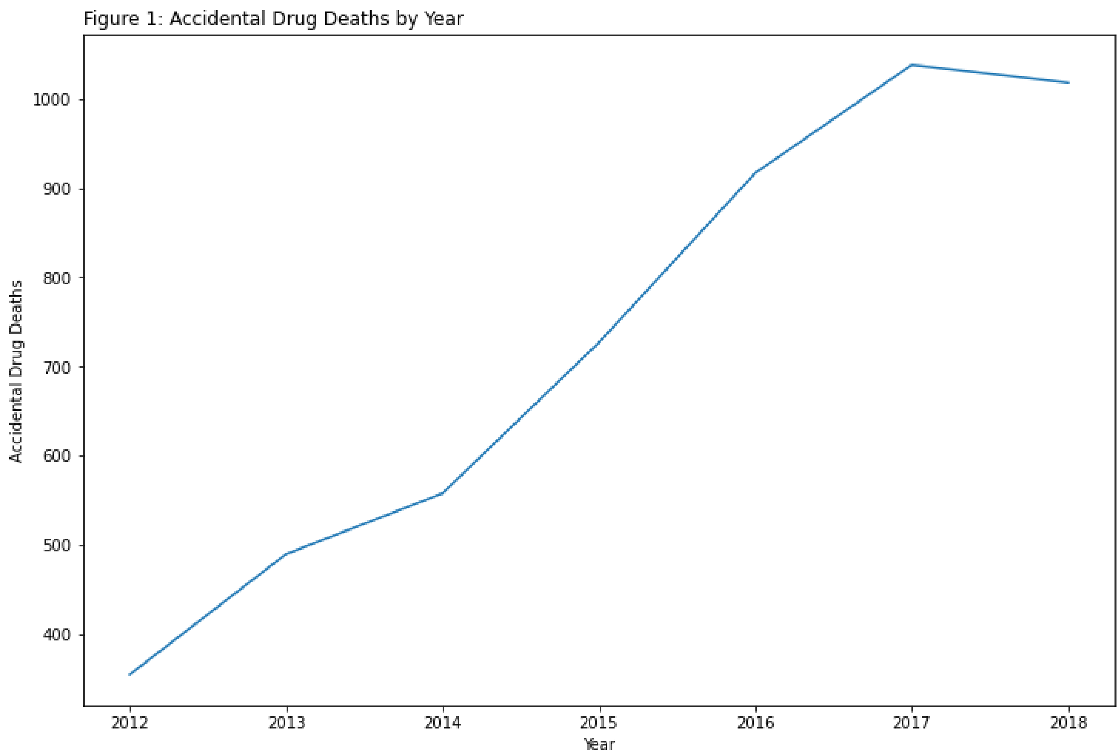
[2]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 5105 entries, 14-0273 to 16-0637
Data columns (total 43 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   Date                 5103 non-null   object
 1   Year                 5103 non-null   float64
 2   DateType             5103 non-null   object
 3   Age                  5102 non-null   float64
 4   Sex                  5099 non-null   object
 5   Race                 5092 non-null   object
 6   ResidenceCity        4932 non-null   object
 7   ResidenceCounty      4308 non-null   object
 8   ResidenceState       3556 non-null   object
 9   DeathCity            5100 non-null   object
 10  DeathCounty          4005 non-null   object
 11  Location             5081 non-null   object
 12  LocationifOther      590 non-null    object
 13  DescriptionofInjury  4325 non-null   object
 14  InjuryPlace          5039 non-null   object
 15  InjuryCity           3349 non-null   object
 16  InjuryCounty         2364 non-null   object
 17  InjuryState          1424 non-null   object
 18  COD                  5105 non-null   object
 19  OtherSignifican      169 non-null    object
 20  Heroin               5105 non-null   int64
 21  Cocaine              5105 non-null   int64
```

```
 22  Fentanyl             5105 non-null   int64
 23  FentanylAnalogue     5105 non-null   int64
 24  Oxycodone            5105 non-null   int64
 25  Oxymorphone          5105 non-null   int64
 26  Ethanol              5105 non-null   int64
 27  Hydrocodone          5105 non-null   int64
 28  Benzodiazepine       5105 non-null   int64
 29  Methadone            5105 non-null   int64
 30  Amphet               5105 non-null   int64
 31  Tramad               5105 non-null   int64
 32  Morphine_NotHeroin   5105 non-null   int64
 33  Hydromorphone        5105 non-null   int64
 34  Other                5105 non-null   int64
 35  OpiateNOS            5105 non-null   int64
 36  AnyOpioid            5105 non-null   int64
 37  Medication           5105 non-null   int64
 38  Number_of_drugs      5105 non-null   int64
 39  MannerofDeath        5095 non-null   object
 40  DeathCityGeo         5105 non-null   object
 41  ResidenceCityGeo     5012 non-null   object
 42  InjuryCityGeo        5027 non-null   object
dtypes: float64(2), int64(19), object(22)
memory usage: 1.7+ MB
```

Let's first visualize the total number of deaths

[3]:
```
df.groupby(['Year'])['Date'].count().plot(figsize=(12,8))
plt.title('Figure 1: Accidental Drug Deaths by Year', loc = 'left')
plt.ylabel('Accidental Drug Deaths')
plt.show()
```




And now let's look at the most common drugs

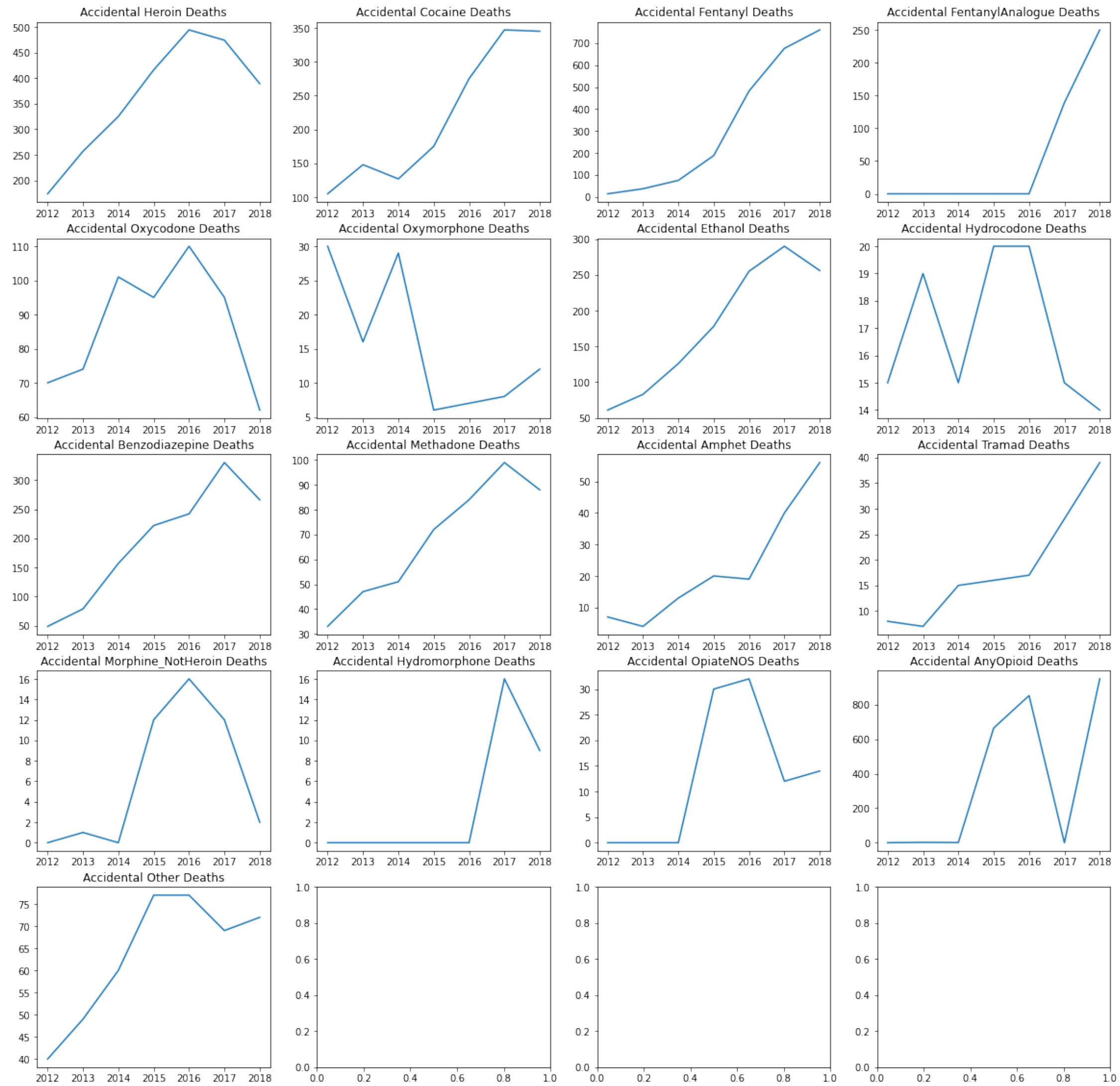
[4]:
```python
drug_columns = ['Heroin', 'Cocaine', 'Fentanyl', 'FentanylAnalogue',␣
↪'Oxycodone', 'Oxymorphone', 'Ethanol', 'Hydrocodone',
                'Benzodiazepine', 'Methadone', 'Amphet', 'Tramad',␣
↪'Morphine_NotHeroin', 'Hydromorphone', 'OpiateNOS', 'AnyOpioid', 'Other']
df[drug_columns].sum().sort_values(ascending=False).astype(int)
```

[4]:
```
Heroin                2529
AnyOpioid             2471
Fentanyl              2232
Cocaine               1523
Benzodiazepine        1345
Ethanol               1249
Oxycodone              607
Methadone              474
Other                  445
FentanylAnalogue       389
Amphet                 159
Tramad                 130
Hydrocodone            118
Oxymorphone            108
OpiateNOS               88
```

```
Morphine_NotHeroin      43
Hydromorphone           25
dtype: int32
```

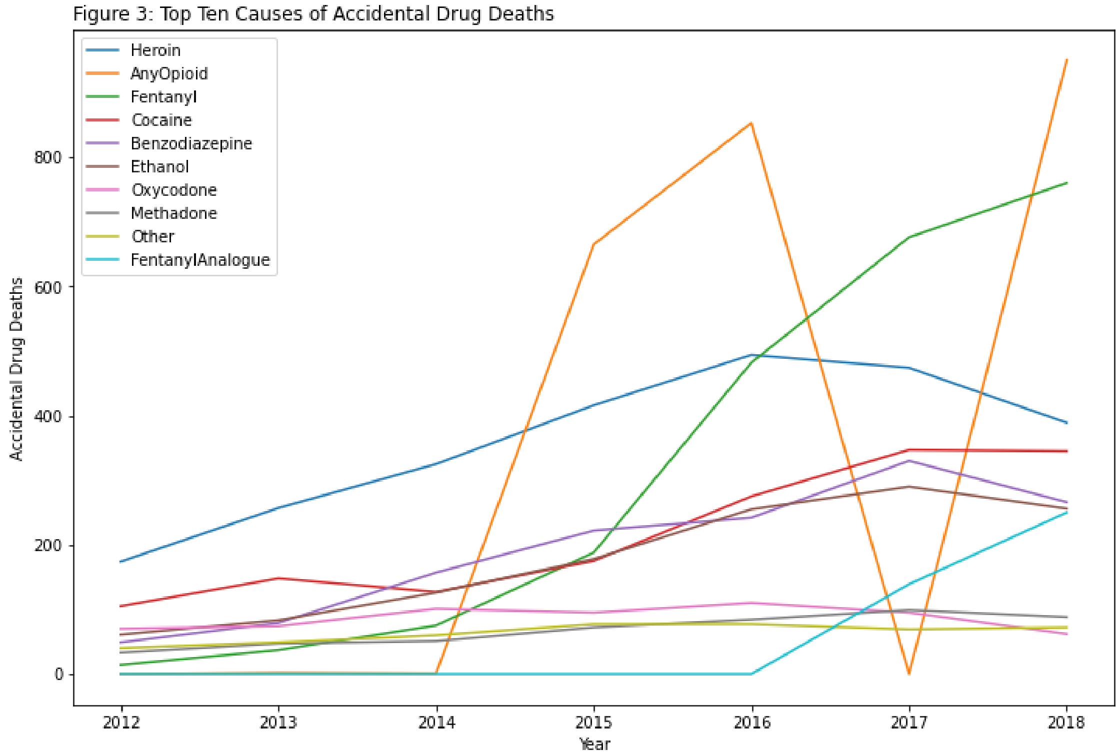
This tells us that Heroin, AnyOpioid and Fentanyl were the top three causes of accidental drug deaths from 2012-2018, but it gives us no sense of yearly change.

[5]:
```
fig, ax = plt.subplots(nrows=5, ncols=4, figsize=(20,20))
drug = 0
for i in range(5):
    if i != 4:
        for j in range(4):
            ax[i][j].plot(df.groupby(['Year'])[drug_columns[drug]].sum())
            ax[i][j].set_title('Accidental ' + str(drug_columns[drug]) + '␣
↪Deaths')
            drug +=1
    else:
        ax[i][0].plot(df.groupby(['Year'])[drug_columns[drug]].sum())
        ax[i][0].set_title('Accidental ' + str(drug_columns[drug]) + ' Deaths')
```



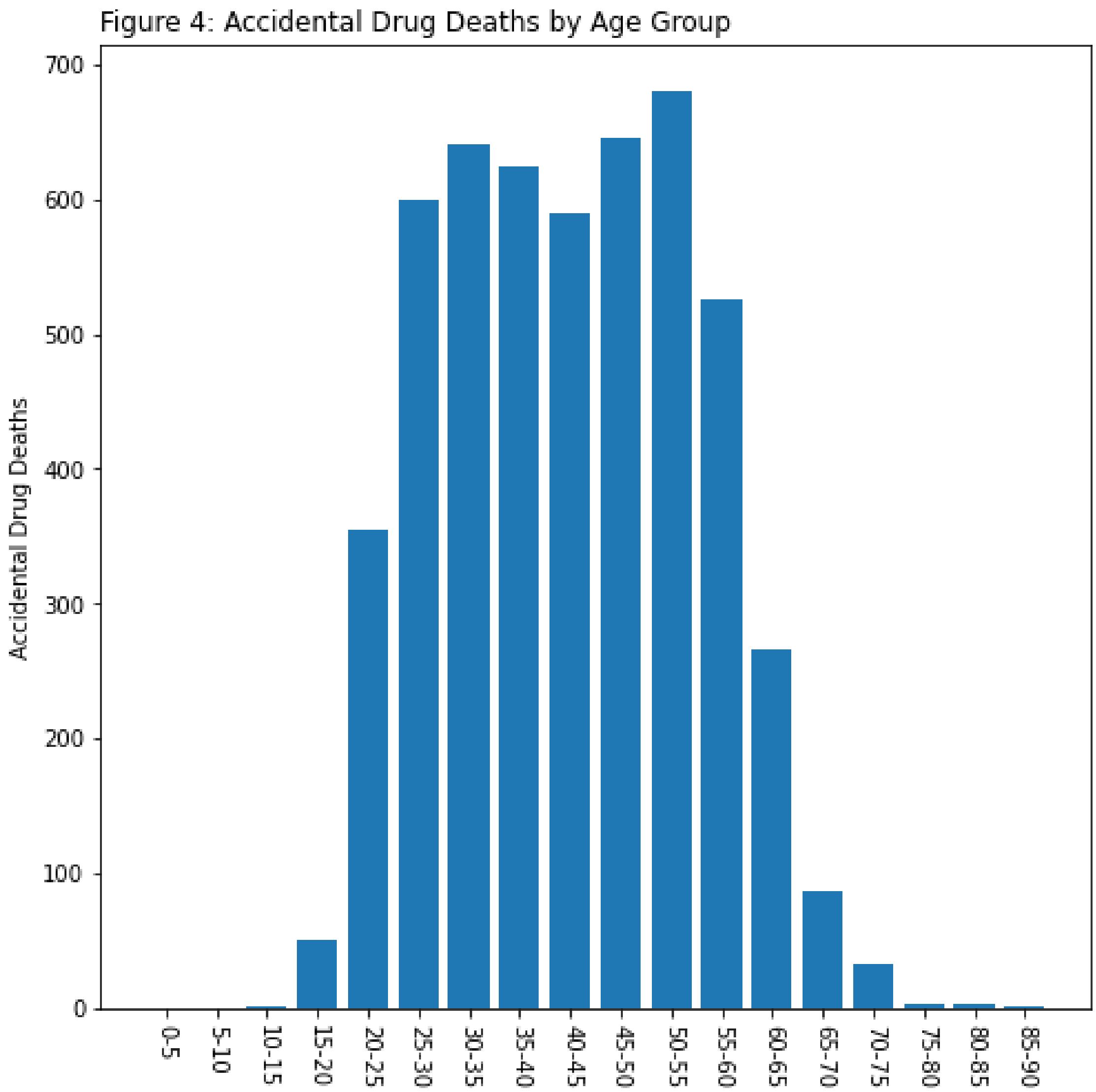

This graph let's us see any trends in the data. Many of the drugs (Heroin, Cocaine, Ethanol, Benzodiazepines and Methadone seem to have peaked in 2016 or 2017 and have declined since. We also see the rapid rise in drug deaths related to Fentanyl and AnyOpioids. To get a better sense of scale and time, let's plot the 10 most common drugs against each other.

[6]:
```
top_ten = df.groupby(['Year'])[drug_columns].sum().sum().
 ↪sort_values(ascending=False).head(10).index.tolist()
df.groupby(['Year'])[top_ten].sum().plot(figsize=(12,8))
plt.title('Figure 3: Top Ten Causes of Accidental Drug Deaths', loc = 'left')
plt.ylabel('Accidental Drug Deaths')
plt.show()
```




We see that Heroin was responsible for the most deaths in 2012, but in 2018 it is now the third most common with AnyOpioid and Fentanyl taking its place. We also see a sharp increase in AnyOpioid deaths from 0 in 2014 to over 600 in 2015, but this is likely due to how Connecticut classified Opioid deaths before 2015. Similarly, the decrease to 0 Opioid deaths in 2017 is likely due to changes in documentation or data entry. Unfortunately, looking at Figure 2 we see no other drug that had a large increase in 2017 and so there is no simple way to impute the data.

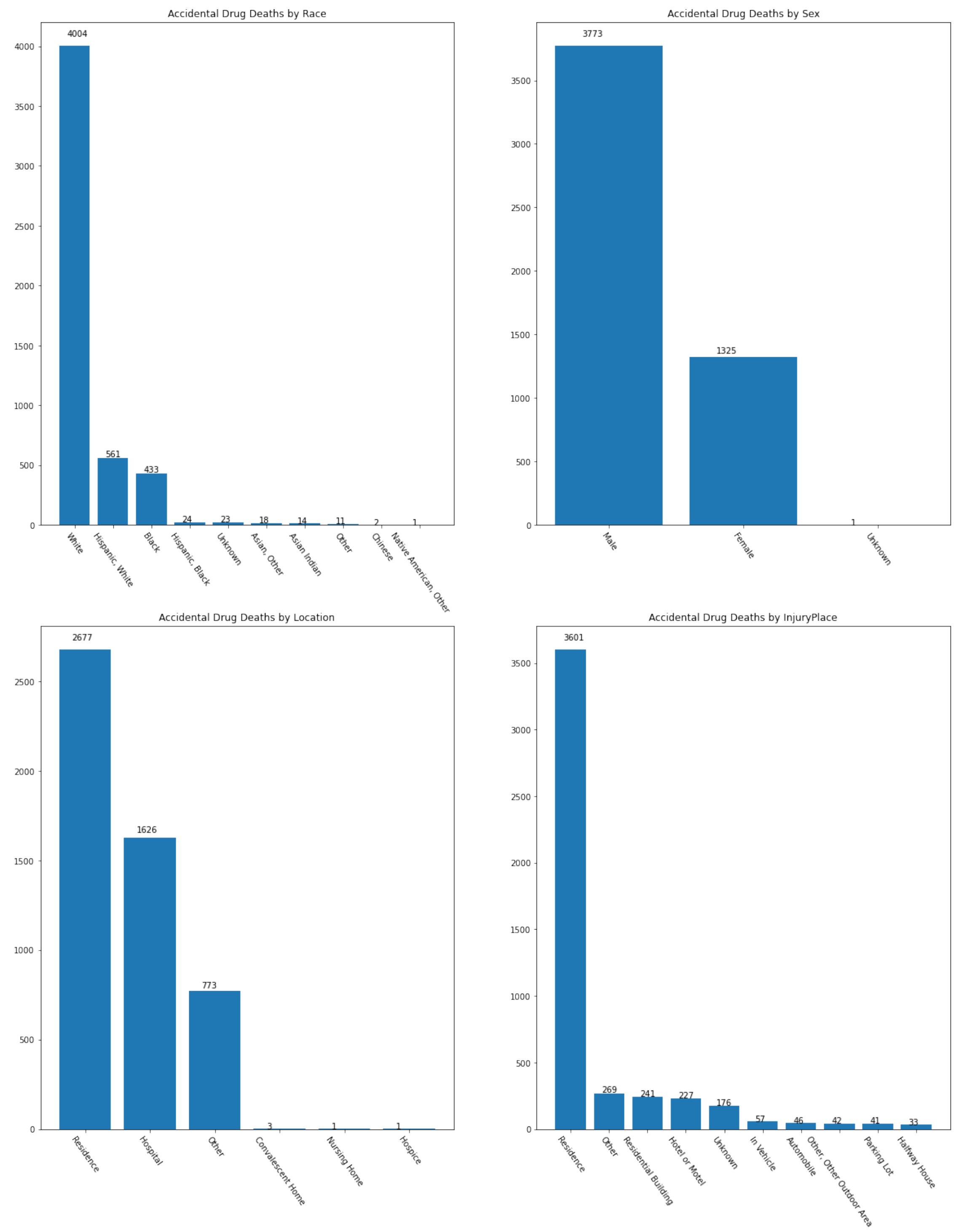
[7]:
```
bins = []
bins.append(0)
for i in range(0, 91):
    if i % 5 == 4:
        bins.append(i)
labels = ['0-5', '5-10', '10-15', '15-20', '20-25', '25-30', '30-35', '35-40',␣
↪'40-45', '45-50', '50-55', '55-60', '60-65', '65-70', '70-75', '75-80',␣
↪'80-85', '85-90']
df['Age Bracket'] = pd.cut(df['Age'], bins = bins, labels = labels)
age_brackets = df.groupby(['Age Bracket'])['Date'].count()
fig, ax = plt.subplots(figsize=(8,8))
plt.bar(age_brackets.index, age_brackets)
plt.xticks(rotation=270)
plt.title('Figure 4: Accidental Drug Deaths by Age Group', loc = 'left')
plt.ylabel('Accidental Drug Deaths')
plt.show()
```




By separating ages into bins we get a less noisy display of the data. This way, we see that the vast majority of accidental drug deaths occur in people between the ages of 20 and 65. There are also two humps in the data at 25-30 and at 50-55. This might be noise, but could reflect differences in which drugs are common for each age group.

[8]:
```
description_columns = ['Race','Sex', 'Location',  'InjuryPlace']
fig, ax = plt.subplots(nrows=2, ncols=2, figsize=(20,25))
counter = 0
for row in range(2):
    for column in range(2):
        temp_df = df[description_columns[counter]].value_counts().head(10)
        ax[row][column].bar(temp_df.index, temp_df.values)
        ax[row][column].set_xticklabels(labels = temp_df.index, rotation = 305)
```

```
        ax[row][column].set_title('Accidental Drug Deaths by ' +␣
↪str(description_columns[counter]))
        for t, v in enumerate(temp_df):
            ax[row][column].text(t-.2, temp_df.values[t].max()*1.02, v)
        counter +=1
```



These graphs show some summary data. We see that most victims are white, followed by Hispanic, White and then Black. Most of the victims are also men. The location of death is typically in someone's residence or a hospital, and the majority of locations of injury are in someone's residence or lodging.

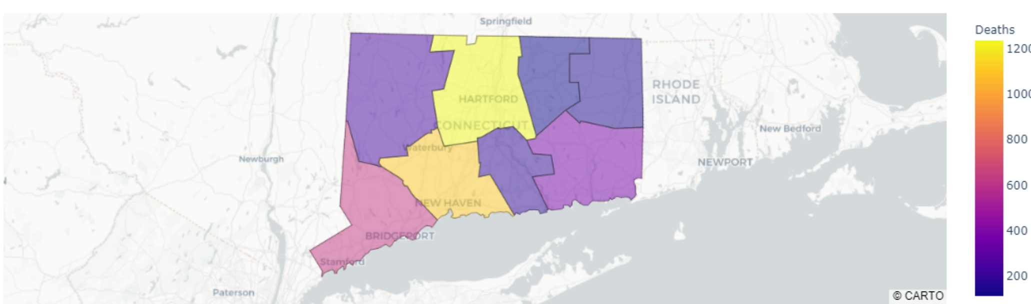
[9]:
```
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/geojson-counties-fips.json') as response:
    counties = json.load(response)

fips = ['09001', '09003', '09005', '09007', '09009', '09011', '09013', '09015']

map_data = pd.DataFrame(df.groupby(['DeathCounty'])['Date'].count().tolist(), fips)
map_data = map_data.reset_index()
map_data.columns = ['FIPS', 'Deaths']

fig = px.choropleth_mapbox(map_data, geojson=counties, locations='FIPS', color='Deaths',
                           mapbox_style="carto-positron",
                           zoom=7, center = {"lat":41.5 , "lon": -72.7658},
                           opacity=0.5
                          )

fig.show()
```




Since we have county level data on deaths, we use plotly to show where most accidental drug deaths occur. The most common county is Hartford (middle top in yellow) followed by New Haven (middle bottom in orange) and Fairfield (bottom left). Meanwhile, Litchfield (top left), Tolland (top middle), Windham (top right), Middlesex (middle bottom right) and New London (bottom right) have relatively few deaths.

Next we look at combinations of drugs

[10]:
```
top_five = df.groupby(['Year'])[drug_columns].sum().sum().
↪sort_values(ascending=False).head(5).index.tolist()

for drug in top_five:
    print('Percentage of ' + drug + ' Deaths Where User also had _____ in their␣
↪system')
    drug_i_deaths = df[df[drug] == 1]
    other_drugs = list(set(drug_columns) - {drug})
    sorted_most_associated_drugs = drug_i_deaths[other_drugs].sum().
↪sort_values(ascending=False).head(5)
    print(sorted_most_associated_drugs.div(len(drug_i_deaths)))
```

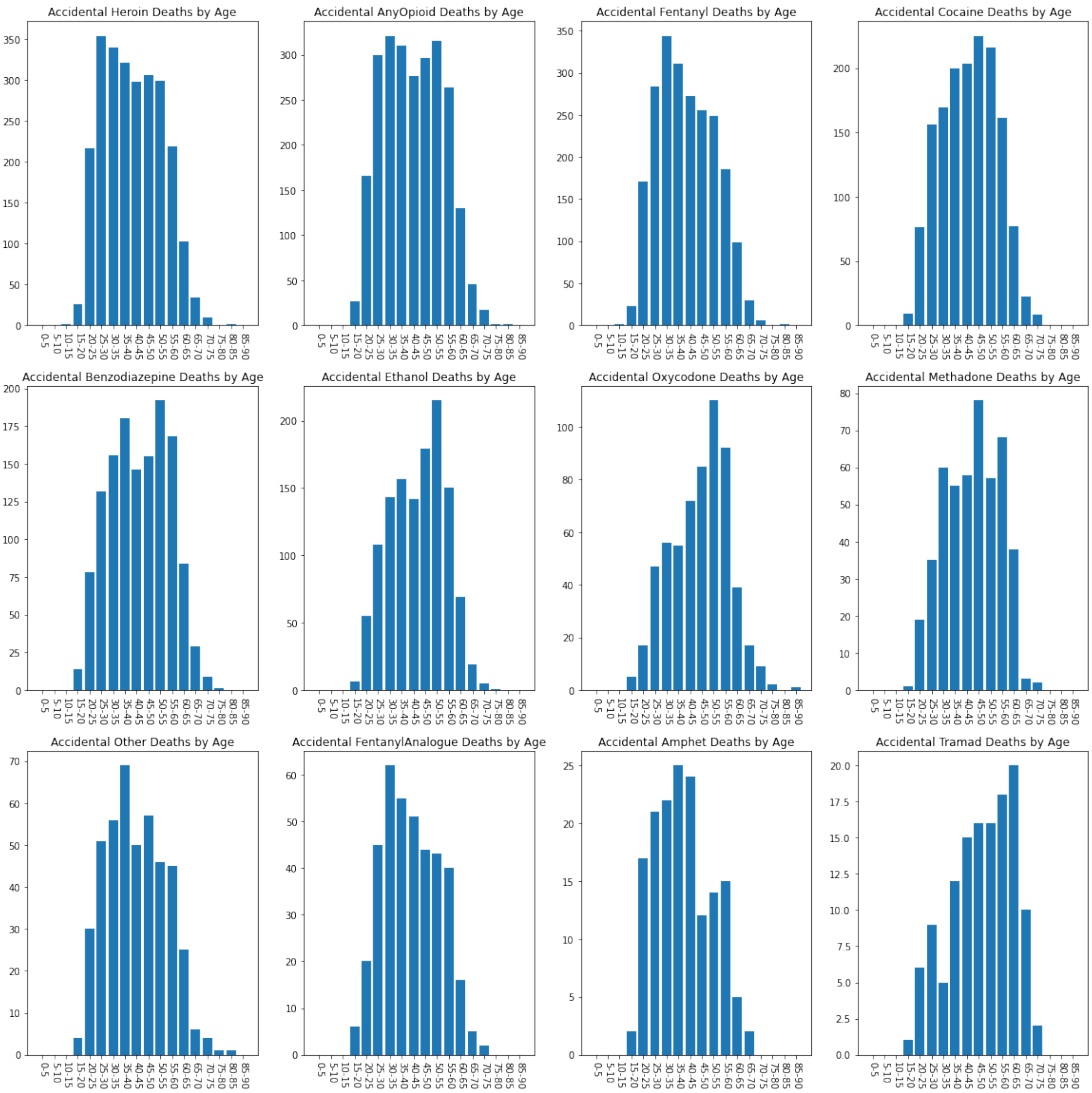
```
Percentage of Heroin Deaths Where User also had _____ in their system
AnyOpioid         0.511665
Fentanyl          0.419533
Cocaine           0.296955
Ethanol           0.233294
Benzodiazepine    0.216686
dtype: float64
Percentage of AnyOpioid Deaths Where User also had _____ in their system
Fentanyl          0.576285
Heroin            0.523675
Benzodiazepine    0.276811
Cocaine           0.276406
Ethanol           0.248482
dtype: float64
Percentage of Fentanyl Deaths Where User also had _____ in their system
AnyOpioid         0.637993
Heroin            0.475358
Cocaine           0.315860
Ethanol           0.242384
Benzodiazepine    0.238799
dtype: float64
Percentage of Cocaine Deaths Where User also had _____ in their system
Heroin            0.493106
Fentanyl          0.462902
AnyOpioid         0.448457
Ethanol           0.214708
Benzodiazepine    0.176625
dtype: float64
Percentage of Benzodiazepine Deaths Where User also had _____ in their system
AnyOpioid    0.508550
Heroin       0.407435
Fentanyl     0.396283
Ethanol      0.241636
Cocaine      0.200000
```

```
dtype: float64
```

Heuristically we can look at this data to get a vague understanding of which combinations of drugs are most lethal. For example, see that 51% of all Heroine deaths also had Opioids in their system. The largest percentage is Fentanyl deaths, where 64% of Fentanyl deaths were also accompanied with Opioids.

[13]:
```
top_twelve = df.groupby(['Year'])[drug_columns].sum().sum().
 ↪sort_values(ascending=False).head(12).index.tolist()

grouped_drugs = df.groupby(['Age Bracket'])[top_twelve].sum()
grouped_drugs_column_list = grouped_drugs.columns.tolist()
fig, ax = plt.subplots(nrows=3, ncols=4, figsize=(20,20))
drug = 0
for row in range(3):
        for col in range(4):
            ax[row][col].bar(grouped_drugs.index,␣
 ↪grouped_drugs[grouped_drugs_column_list[drug]])
            ax[row][col].set_title('Accidental ' +␣
 ↪str(grouped_drugs_column_list[drug]) + ' Deaths by Age')
            ax[row][col].set_xticklabels(labels = grouped_drugs.index, rotation␣
 ↪= 270)
            drug +=1
```


Accidental Heroin Deaths by Age
Accidental AnyOpioid Deaths by Age
Accidental Fentanyl Deaths by Age
Accidental Cocaine Deaths by Age
Accidental Benzodiazepine Deaths by Age
Accidental Ethanol Deaths by Age
Accidental Oxycodone Deaths by Age
Accidental Methadone Deaths by Age
Accidental Other Deaths by Age
Accidental FentanylAnalogue Deaths by Age
Accidental Amphet Deaths by Age
Accidental Tramad Deaths by Age

[ ]: