ata-analysis-on-gun-violence-in-us

November 15, 2024

1 Data Analysis on Gun Violence in US

1.0.1 About Dataset

- Gun Violence Dataset Overview (Mass Shootings in 2024)
- This dataset provides an in-depth look at mass shootings across the United States in 2024, up until October 20th, sourced from the Gun Violence Archive.
- It captures essential details such as incident ID, date, state, city, victims (killed and injured), and suspects involved.
- Additionally, geographical coordinates are included to allow for spatial analysis of gun violence trends.

1.0.2 Key Features of the Dataset

- Incident IDs: A unique identifier for each incident, categorized and grouped based on their range.
- Incident Date: Captures the time period over which the incidents occurred, broken down into monthly intervals.
- State and City/County: A breakdown of incidents across various states and cities, identifying regions with the most gun violence activity.
- Victims Killed and Injured: Provides a count of victims in each incident, allowing for the analysis of both fatal and non-fatal outcomes.
- Suspects Killed, Injured, and Arrested: Captures data related to suspects involved in the incidents, including their status post-incident (e.g., killed, injured, or arrested).
- Latitude and Longitude: Geographical coordinates for the locations of the incidents, enabling mapping and geospatial analysis.
- Coordinates Found: Indicates whether valid geographical coordinates were found for each incident.

[1]:
```
#Loading Libraries

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

[2]:
```
#loading dataset
df = pd.read_csv("Dataset/gun.csv")
```

[3]:
```python
df.head()
```

[3]:
```
   Incident ID      Incident Date         State City Or County  \
0      3052758  October 21, 2024    Washington       Fall City
1      3052028  October 20, 2024     Tennessee         Jackson
2      3051984  October 20, 2024     Louisiana     Baton Rouge
3      3051041  October 19, 2024  Pennsylvania    Philadelphia
4      3050940  October 19, 2024   Mississippi       Lexington

                             Address  Victims Killed  Victims Injured  \
0      7700 block of Lake Alice Rd SE               5                1
1                  2310 N Highland Ave               1                8
2  9700 block of Greenwell Springs Rd               0                5
3                    2517 N Jessup St               0                7
4                          24904 MS-17               3                8

   Suspects Killed  Suspects Injured  Suspects Arrested  Operations  Latitude  \
0                0                 0                  1         NaN  47.56812
1                0                 0                  0         NaN  35.61390
2                0                 0                  0         NaN  30.44335
3                0                 0                  0         NaN  39.95222
4                0                 0                  0         NaN  33.11464

   Longitude Coordinates_Found
0 -121.89086               Yes
1  -88.81940               Yes
2  -91.18664               Yes
3  -75.16218               Yes
4  -90.05281               Yes
```

[4]:
```python
df.columns
```

[4]:
```
Index(['Incident ID', 'Incident Date', 'State', 'City Or County', 'Address',
       'Victims Killed', 'Victims Injured', 'Suspects Killed',
       'Suspects Injured', 'Suspects Arrested', 'Operations', 'Latitude',
       'Longitude', 'Coordinates_Found'],
      dtype='object')
```

[5]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 427 entries, 0 to 426
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Incident ID        427 non-null    int64
 1   Incident Date      427 non-null    object
```

```
 2   State              427 non-null    object
 3   City Or County     427 non-null    object
 4   Address            426 non-null    object
 5   Victims Killed     427 non-null    int64
 6   Victims Injured    427 non-null    int64
 7   Suspects Killed    427 non-null    int64
 8   Suspects Injured   427 non-null    int64
 9   Suspects Arrested  427 non-null    int64
 10  Operations         0 non-null      float64
 11  Latitude           427 non-null    float64
 12  Longitude          427 non-null    float64
 13  Coordinates_Found  427 non-null    object
dtypes: float64(3), int64(6), object(5)
memory usage: 46.8+ KB
```

[6]:
```
df.describe()
```

[6]:
```
        Incident ID  Victims Killed  Victims Injured  Suspects Killed  \
count  4.270000e+02      427.000000       427.000000       427.000000
mean   2.929432e+06        1.000000         4.437939         0.070258
std    6.876031e+04        1.210285         2.849555         0.264896
min    2.791411e+06        0.000000         0.000000         0.000000
25%    2.879320e+06        0.000000         3.000000         0.000000
50%    2.939030e+06        1.000000         4.000000         0.000000
75%    2.982807e+06        1.000000         5.000000         0.000000
max    3.052758e+06        8.000000        28.000000         2.000000

       Suspects Injured  Suspects Arrested  Operations    Latitude   Longitude
count        427.000000         427.000000         0.0  427.000000  427.000000
mean           0.063232           0.711944         NaN   36.941819  -88.791959
std            0.271030           1.178414         NaN    4.934289   12.976850
min            0.000000           0.000000         NaN   21.449910 -166.739450
25%            0.000000           0.000000         NaN   33.449545  -91.910580
50%            0.000000           0.000000         NaN   37.687490  -86.811790
75%            0.000000           1.000000         NaN   40.692450  -80.735325
max            2.000000           7.000000         NaN   68.349440  -70.256650
```

[8]:
```
df.isna().sum()
```

[8]:
```
Incident ID          0
Incident Date        0
State                0
City Or County       0
Address              1
Victims Killed       0
Victims Injured      0
Suspects Killed      0
```

```
Suspects Injured        0
Suspects Arrested       0
Operations            427
Latitude                0
Longitude               0
Coordinates_Found       0
dtype: int64
```

[9]:
```python
df.shape
```

[9]:
```
(427, 14)
```

[12]:
```python
df.drop(df[['Address','Operations']], axis=1,inplace=True)
```

[13]:
```python
df.head()
```

[13]:
```
   Incident ID      Incident Date         State City Or County  Victims Killed  \
0      3052758  October 21, 2024    Washington       Fall City               5
1      3052028  October 20, 2024     Tennessee         Jackson               1
2      3051984  October 20, 2024     Louisiana     Baton Rouge               0
3      3051041  October 19, 2024  Pennsylvania    Philadelphia               0
4      3050940  October 19, 2024   Mississippi       Lexington               3

   Victims Injured  Suspects Killed  Suspects Injured  Suspects Arrested  \
0                1                0                 0                  1
1                8                0                 0                  0
2                5                0                 0                  0
3                7                0                 0                  0
4                8                0                 0                  0

   Latitude  Longitude Coordinates_Found
0  47.56812 -121.89086               Yes
1  35.61390  -88.81940               Yes
2  30.44335  -91.18664               Yes
3  39.95222  -75.16218               Yes
4  33.11464  -90.05281               Yes
```

[15]:
```python
df.isna().sum()
```

[15]:
```
Incident ID          0
Incident Date        0
State                0
City Or County       0
Victims Killed       0
Victims Injured      0
Suspects Killed      0
Suspects Injured     0
```

```
Suspects Arrested    0
Latitude             0
Longitude            0
Coordinates_Found    0
dtype: int64
```

[16]:
```
df.shape
```

[16]:
```
(427, 12)
```

1. What is the total number of incidents in the dataset?

[17]:
```
total_incidents = df.shape[0]
print(f"Total number of incidents: {total_incidents}")
```

```
Total number of incidents: 427
```

[ ]:

2.State with the highest number of incidents

[19]:
```
highest_incidents = df['State'].value_counts().idxmax()
print(f"State with the highest number of incidents: {highest_incidents}")
```

```
State with the highest number of incidents: Illinois
```

[ ]:

3. Total number of victims killed and injured

[23]:
```
total_victims_killed = df['Victims Killed'].sum()
total_victims_injured = df['Victims Injured'].sum()
print(f"Total victims killed: {total_victims_killed},\nTotal victims injured:␣
↪{total_victims_injured}")
```

```
Total victims killed: 427,
Total victims injured: 1895
```

[ ]:

4. Average number of suspects arrested per incident

[24]:
```
average_suspects_arrested = df['Suspects Arrested'].mean()
print(f"Average suspects arrested per incident: {average_suspects_arrested:.
↪2f}")
```

```
Average suspects arrested per incident: 0.71
```

[ ]:

5. **Number of incidents by year**

[26]:
```
df['Incident Year'] = pd.to_datetime(df['Incident Date']).dt.year
incidents_by_year = df['Incident Year'].value_counts()
print("Number of incidents by year: ")
print(incidents_by_year)
```

```
Number of incidents by year:
2024    427
Name: Incident Year, dtype: int64
```

[ ]:
```

```

6. **Coordinates of the incident with the highest total victims (killed + injured)**

[27]:
```
df['Total Victims'] = df['Victims Killed'] + df['Victims Injured']
most_affected_coordinates = df.loc[df['Total Victims'].idxmax(), ['Latitude',␣
 ↪'Longitude']]
print("Coordinates of the most affected incident:")
print(most_affected_coordinates)
```

```
Coordinates of the most affected incident:
Latitude     41.08431
Longitude   -81.51431
Name: 233, dtype: object
```

[ ]:
```

```

1.1 Data Visualization using Mayavi and python

[ ]:
```

```

[29]:
```
!pip install mayavi
```

```
Collecting mayavi
  Downloading mayavi-4.8.2.tar.gz (7.1 MB)
     ---------------------------------------- 0.0/7.1 MB ? eta -:--:--
     ---------------------------------------- 0.0/7.1 MB 682.7 kB/s eta 0:00:11
     - -------------------------------------- 0.3/7.1 MB 3.5 MB/s eta 0:00:02
     -------- ------------------------------- 1.5/7.1 MB 11.0 MB/s eta 0:00:01
     --------------- ------------------------ 2.8/7.1 MB 16.0 MB/s eta 0:00:01
     ------------------ --------------------- 3.2/7.1 MB 14.7 MB/s eta 0:00:01
     ------------------------ --------------- 4.4/7.1 MB 15.5 MB/s eta 0:00:01
     ----------------------------- ---------- 5.2/7.1 MB 16.5 MB/s eta 0:00:01
     ----------------------------------- ---- 6.3/7.1 MB 17.5 MB/s eta 0:00:01
     ---------------------------------------- 7.1/7.1 MB 17.5 MB/s eta 0:00:00
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
```

```
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Collecting apptools (from mayavi)
  Downloading apptools-5.3.0-py3-none-any.whl.metadata (4.3 kB)
Collecting envisage (from mayavi)
  Downloading envisage-7.0.3-py3-none-any.whl.metadata (5.2 kB)
Requirement already satisfied: numpy in c:\users\user\anaconda3\lib\site-
packages (from mayavi) (1.24.4)
Collecting pyface>=6.1.1 (from mayavi)
  Downloading pyface-8.0.0-py3-none-any.whl.metadata (7.7 kB)
Requirement already satisfied: pygments in c:\users\user\anaconda3\lib\site-
packages (from mayavi) (2.15.1)
Collecting traits>=6.0.0 (from mayavi)
  Downloading traits-6.4.3-cp311-cp311-win_amd64.whl.metadata (5.2 kB)
Collecting traitsui>=7.0.0 (from mayavi)
  Downloading traitsui-8.0.0-py3-none-any.whl.metadata (6.8 kB)
Requirement already satisfied: packaging in c:\users\user\anaconda3\lib\site-
packages (from mayavi) (23.1)
Collecting vtk (from mayavi)
  Using cached vtk-9.3.1-cp311-cp311-win_amd64.whl.metadata (5.3 kB)
Requirement already satisfied: setuptools in c:\users\user\anaconda3\lib\site-
packages (from envisage->mayavi) (68.2.2)
Requirement already satisfied: matplotlib>=2.0.0 in
c:\users\user\anaconda3\lib\site-packages (from vtk->mayavi) (3.8.0)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\user\anaconda3\lib\site-
packages (from matplotlib>=2.0.0->vtk->mayavi) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(1.4.4)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(10.2.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\user\anaconda3\lib\site-packages (from matplotlib>=2.0.0->vtk->mayavi)
(2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\user\anaconda3\lib\site-
packages (from python-dateutil>=2.7->matplotlib>=2.0.0->vtk->mayavi) (1.16.0)
```

```
Downloading pyface-8.0.0-py3-none-any.whl (1.3 MB)
   ---------------------------------------- 0.0/1.3 MB ? eta -:--:--
   ------------------------------ --------- 1.0/1.3 MB 31.4 MB/s eta 0:00:01
   ---------------------------------------- 1.3/1.3 MB 16.6 MB/s eta 0:00:00
Downloading traits-6.4.3-cp311-cp311-win_amd64.whl (5.0 MB)
   ---------------------------------------- 0.0/5.0 MB ? eta -:--:--
   -------- ------------------------------- 1.1/5.0 MB 34.6 MB/s eta 0:00:01
   ----------- ---------------------------- 1.4/5.0 MB 17.9 MB/s eta 0:00:01
   ----------------- ---------------------- 2.2/5.0 MB 17.7 MB/s eta 0:00:01
   ----------------------------- ---------- 3.7/5.0 MB 21.5 MB/s eta 0:00:01
   ---------------------------------------  5.0/5.0 MB 22.8 MB/s eta 0:00:01
   ---------------------------------------- 5.0/5.0 MB 21.3 MB/s eta 0:00:00
Downloading traitsui-8.0.0-py3-none-any.whl (1.5 MB)
   ---------------------------------------- 0.0/1.5 MB ? eta -:--:--
   ---------------------------- ----------- 1.1/1.5 MB 22.7 MB/s eta 0:00:01
   ---------------------------------------- 1.5/1.5 MB 24.4 MB/s eta 0:00:00
Downloading apptools-5.3.0-py3-none-any.whl (230 kB)
   ---------------------------------------- 0.0/230.0 kB ? eta -:--:--
   --------------------------------------- 230.0/230.0 kB 13.7 MB/s eta 0:00:00
Downloading envisage-7.0.3-py3-none-any.whl (268 kB)
   ---------------------------------------- 0.0/268.9 kB ? eta -:--:--
   ---------------------------------------- 268.9/268.9 kB ? eta 0:00:00
Using cached vtk-9.3.1-cp311-cp311-win_amd64.whl (52.5 MB)
Building wheels for collected packages: mayavi
  Building wheel for mayavi (pyproject.toml): started
  Building wheel for mayavi (pyproject.toml): still running…
  Building wheel for mayavi (pyproject.toml): finished with status 'done'
  Created wheel for mayavi: filename=mayavi-4.8.2-py3-none-any.whl size=19034131
sha256=c778decf85c34c2fffa4294d97faf91d3f931a31a2b05dfef338b57a55b6bb2b
  Stored in directory: c:\users\user\appdata\local\pip\cache\wheels\c7\57\c0\4ed
3e25a0d8c0f07072c4d6355cf4639945a8bf88afaf67558
Successfully built mayavi
Installing collected packages: traits, pyface, apptools, vtk, traitsui,
envisage, mayavi
Successfully installed apptools-5.3.0 envisage-7.0.3 mayavi-4.8.2 pyface-8.0.0
traits-6.4.3 traitsui-8.0.0 vtk-9.3.1
```

[31]:
```
!pip install configobj
```

```
Collecting configobj
  Downloading configobj-5.0.9.tar.gz (101 kB)
     ---------------------------------------- 0.0/101.5 kB ? eta -:--:--
     ---- ----------------------------------- 10.2/101.5 kB ? eta -:--:--
     ----------------------------------- --- 92.2/101.5 kB 1.3 MB/s eta 0:00:01
     ------------------------------------ 101.5/101.5 kB 981.5 kB/s eta 0:00:00
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
```

```
  Getting requirements to build wheel: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Building wheels for collected packages: configobj
  Building wheel for configobj (pyproject.toml): started
  Building wheel for configobj (pyproject.toml): finished with status 'done'
  Created wheel for configobj: filename=configobj-5.0.9-py2.py3-none-any.whl
size=35637
sha256=7a3d079eb3b69ee73d7ca275a5d3ea154758fed8ca333d91f5c07b2207a93878
  Stored in directory: c:\users\user\appdata\local\pip\cache\wheels\64\0b\d9\934
7fb191ffdc88f4b0146338d157b9616de47ddfd93cd2481
Successfully built configobj
Installing collected packages: configobj
Successfully installed configobj-5.0.9
```

[32]:

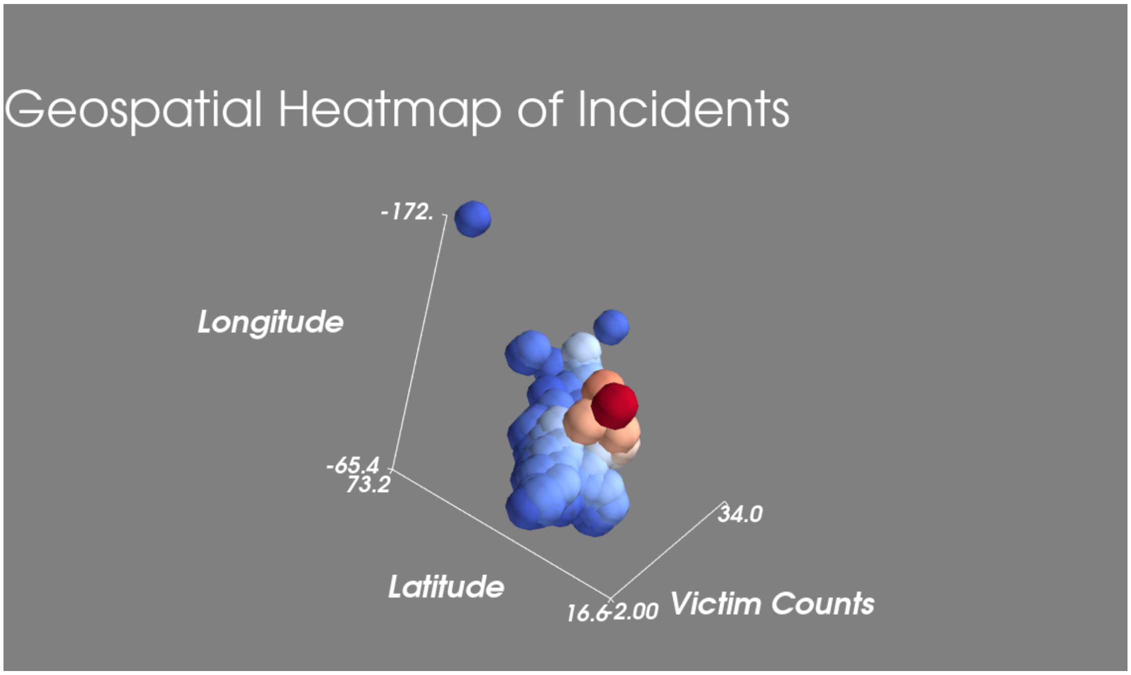
```
from mayavi import mlab
# Geospatial Heatmap
latitudes = df['Latitude']
longitudes = df['Longitude']
victim_counts = df['Victims Killed'] + df['Victims Injured']

# 3D Scatter plot
mlab.figure(size=(800, 600))
mlab.points3d(latitudes, longitudes, victim_counts,
              victim_counts,
              scale_mode='none',
              scale_factor=10,
              colormap='coolwarm')
mlab.axes(xlabel='Latitude', ylabel='Longitude', zlabel='Victim Counts')
mlab.title("Geospatial Heatmap of Incidents")
mlab.show()
```

[39]:

```
from IPython.display import Image, display

# Display an image
image_path = "Images\Geospatial Output.png"
display(Image(filename=image_path))
```



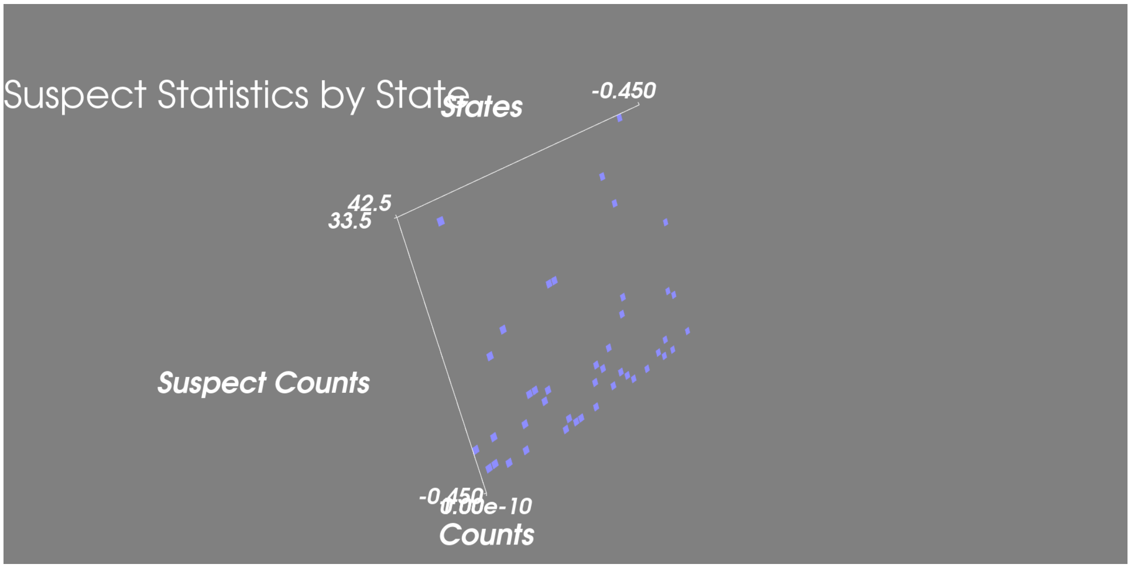
[ ]:

[41]:
```
# Suspect Statistics
states = df['State'].unique()
suspect_counts = df.groupby('State')['Suspects Arrested'].sum()

x = np.arange(len(states))
y = suspect_counts.values

mlab.figure(size=(800, 600))
mlab.barchart(x, y, np.zeros_like(y), colormap='cool', color=(0.5, 0.5, 1.0))
mlab.axes(xlabel='States', ylabel='Suspect Counts', zlabel='Counts')
mlab.title("Suspect Statistics by State")
mlab.show()
```

[40]:
```
# Display an image
image_path = "Images\Suspect Output.png"  # Replace with your image path
display(Image(filename=image_path))
```



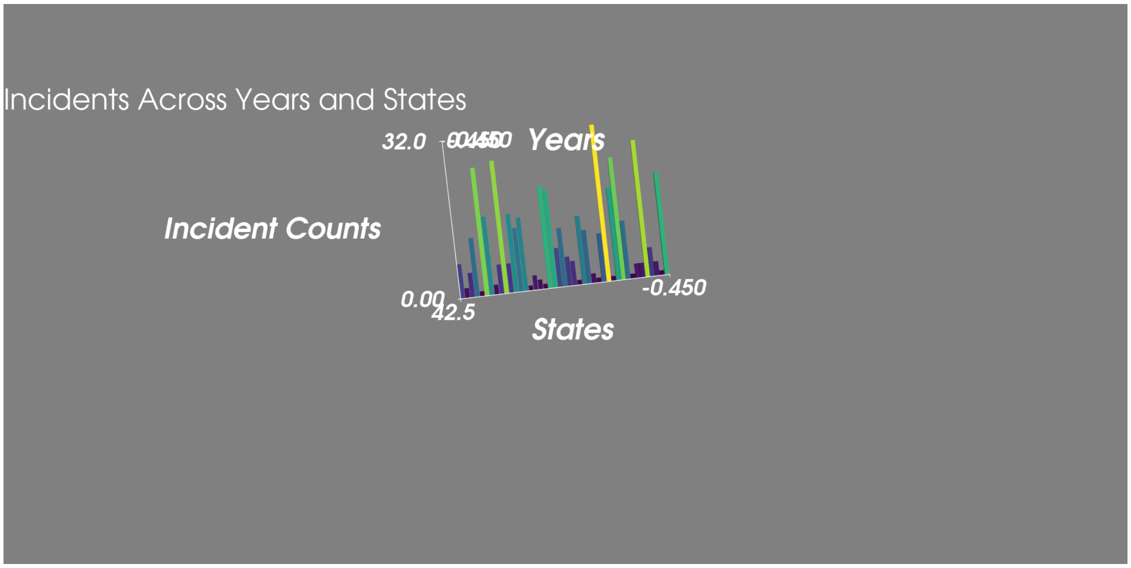
[47]:
```
# Temporal Trend

# Data preparation
years = df['Incident Year'].unique()
states = df['State'].unique()
incident_counts = df.groupby(['Incident Year', 'State']).size().
    ↪unstack(fill_value=0)

x, y = np.meshgrid(range(len(years)), range(len(states)))
z = incident_counts.values.T

# 3D Bar plot
mlab.figure(size=(800, 600))
mlab.barchart(x, y, z, colormap='viridis')
mlab.axes(xlabel='Years', ylabel='States', zlabel='Incident Counts')
mlab.title("Incidents Across Years and States")
mlab.show()
```

[48]:
```
# Display an image
image_path = "Images\incident.png"  # Replace with your image path
display(Image(filename=image_path))
```



[ ]:

[ ]:

- Geospatial Clustering of Incidents: The analysis revealed geographic hotspots where incidents are more frequent, helping identify areas with high criminal activity or risk.
- Impact Analysis by Location: Visualizing the number of victims killed, injured, or suspects arrested across different regions provides insights into the severity of incidents in specific areas.
- Risk Zones Identification: By creating buffers around key locations, we can identify high-risk zones that require increased attention or resources for intervention.
- Region-Based Aggregation: Summarizing incident data by cities, counties, or states helps prioritize areas with the most incidents, guiding law enforcement and resource allocation.
- Effectiveness of Law Enforcement Operations: Mapping the operations and arrests in relation to incidents shows the effectiveness of law enforcement in addressing crime in high-density areas.
- Comparative Analysis of Regions: Comparing incident rates between neighboring regions helps highlight disparities in crime or emergency incidents, offering a foundation for targeted policies.
- Enhanced Decision-Making for Policy Makers: The spatial analysis and visualizations provide valuable insights for policy makers to improve crime prevention strategies and optimize resource distribution.

[ ]: