Visual Analysis of Murder victims in India between 2001 to 2010 using Machine Learning

Alugolu Avinash
Asst Professor
Centurion University of Technology and Management

Karanam Santoshachandra Rao
Asst Professor
Centurion University of Technology and Management

Mudadla Keerthana
Regno:171801120016
Centurion University of Technology and Management

ABSTRACT

Murder victims in India [1] are raises day by day due to some personal, professional or mental reasons. In this paper Authors have analyzed Murder victims between the years 2001 to 2010 using machine learning visualization techniques based on some constrains like: 1) Male and female victims, 2) States having higher murder victims percentage, 3) Total murder victims in the years 2001 to 2010, 4) Statewide murder victims based on age group, 5) Uttar Pradesh female murder victims percentage,6) Total number of murder victims in year 2002, 7) Murder victims growth in India, 8)State wise murder victims of age 30 to 50 years, 9) Murder victims of Two states Bihar and Uttar Pradesh, 10) Minor and major murder victims.

Keywords: Murder, victims, Machine Learning, Visualization, Plotting, Python, Packages, Numpy, Pandas, Seaborn, Matplotlib , Plotly.

Introduction

Analysis of mentioned 10 points are based on python [2] best and top libraries like numpy[3], pandas and visualization libraries like matplotlib[4], plotlly[5]. Generally, Python is high-level general purpose programming and an interpreted language. Developed by Guido van Rossum and primarily released in 1991, Python's design intention emphasizes code readability & understanding with its easy use of significant whitespace or indentation. Its object-oriented approach targets to help developers write unambiguous, very logical code for tiny and large-scale projects.

"Visualization is worth a many thousand words". We are all aware of this expression. This especially applies when we trying to deliver the insight received from the analysis of large datasets. Data visualization plays a major role in the visualization of both tiny and very large-scale data.

One of the important skills of a data scientist or programmer is the ability to express a compelling story, displaying data visually and findings in a possible and stimulating way. Studying how to use a software tool to visualize data-sets will also permits you to fetch information, clear understand the data, and make more perfect decisions.

The main motto of this Data Visualization with Python is to analyze above mentioned 10 points and present that information in the form that makes sense to people. Various techniques have been used for presenting data visually with the help of python libraries namely Matplotlib, Seaborn, and Plotly.

Following python library[6] code is required to import primarily to analyze the data with the help of various function and methods.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.style as style
import matplotlib.ticker as ticker
import matplotlib.animation as animation
from IPython.display import HTML
import plotly.express as px
```

Understanding the dataset

Here, we have considered the data from the dataset[7] “Murder_victim_age_sex.csv” for analysis. Basic idea on dataset is given by following python code.[8]

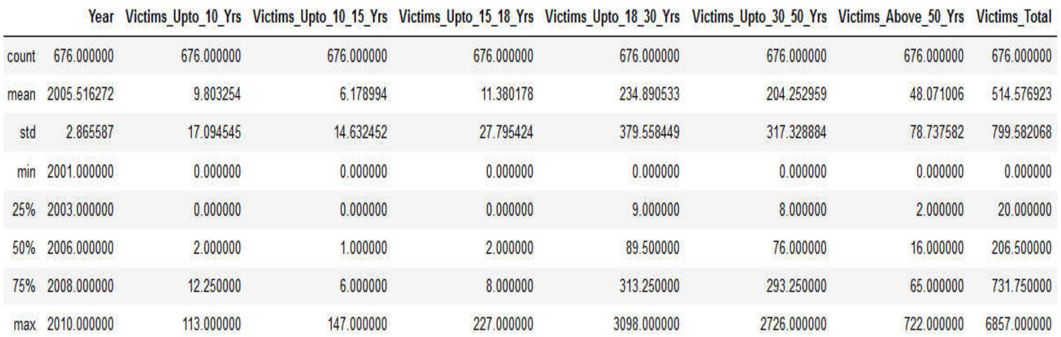
Step1: Reads the Murder_Victim_age_sex.csv file

```
murder_victims=pd.read_csv(r'Murder_victim_age_sex.csv')

murder_victims.head()
```

This above code gives the first 5 rows from the data set[9].

Fig 1 shows the output of the following python code for dataset summarizing the columns.
Code: murder_victims.describe()

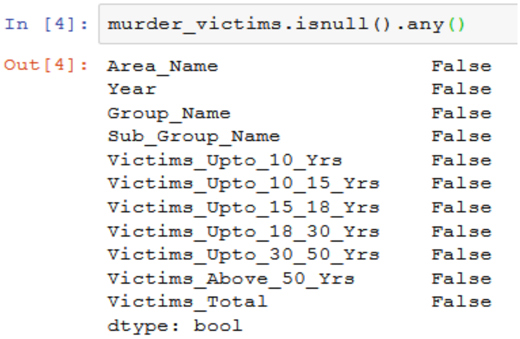


	Year	Victims_Upto_10_Yrs	Victims_Upto_10_15_Yrs	Victims_Upto_15_18_Yrs	Victims_Upto_18_30_Yrs	Victims_Upto_30_50_Yrs	Victims_Above_50_Yrs	Victims_Total
count	676.000000	676.000000	676.000000	676.000000	676.000000	676.000000	676.000000	676.000000
mean	2005.516272	9.803254	6.178994	11.380178	234.890533	204.252959	48.071006	514.576923
std	2.865587	17.094545	14.632452	27.795424	379.558449	317.328884	78.737582	799.582068
min	2001.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2003.000000	0.000000	0.000000	0.000000	9.000000	8.000000	2.000000	20.000000
50%	2006.000000	2.000000	1.000000	2.000000	89.500000	76.000000	16.000000	206.500000
75%	2008.000000	12.250000	6.000000	8.000000	313.250000	293.250000	65.000000	731.750000
max	2010.000000	113.000000	147.000000	227.000000	3098.000000	2726.000000	722.000000	6857.000000

Fig 1: Dataset summarizing the columns

Following python code is used to check for null values in the dataset

Code: murder_victims.isnull().any()



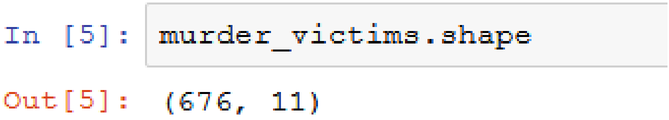
```
In [4]: murder_victims.isnull().any()

Out[4]: Area_Name                  False
        Year                       False
        Group_Name                 False
        Sub_Group_Name             False
        Victims_Upto_10_Yrs        False
        Victims_Upto_10_15_Yrs     False
        Victims_Upto_15_18_Yrs     False
        Victims_Upto_18_30_Yrs     False
        Victims_Upto_30_50_Yrs     False
        Victims_Above_50_Yrs       False
        Victims_Total              False
        dtype: bool
```

Fig 2: Identifying null values in the dataset

Following code is to identify shape of the dataset

Code: murder_victims.shape



```
In [5]: murder_victims.shape

Out[5]: (676, 11)
```

Fig 3: Identifying shape of the dataset

1. Male and Female victims

All plotting can be done with the help of many plotting techniques [10].With the help of these two columns Sub_Group_Name and Victims_Total observed male and female victims total.Fig 4 is its visualization.

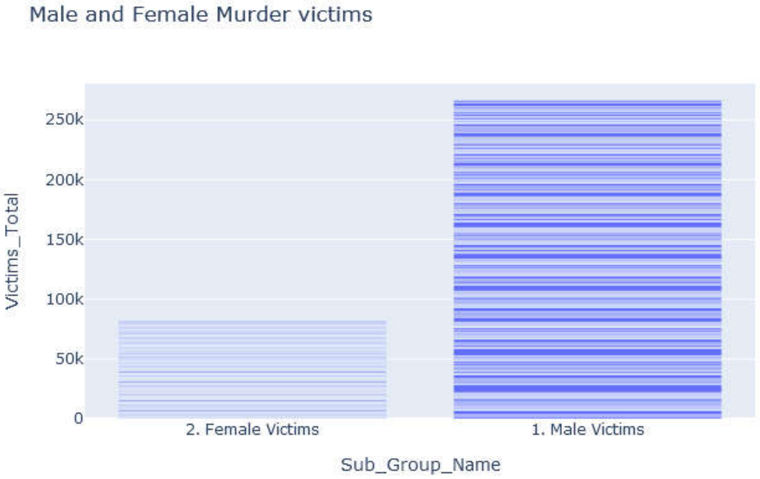
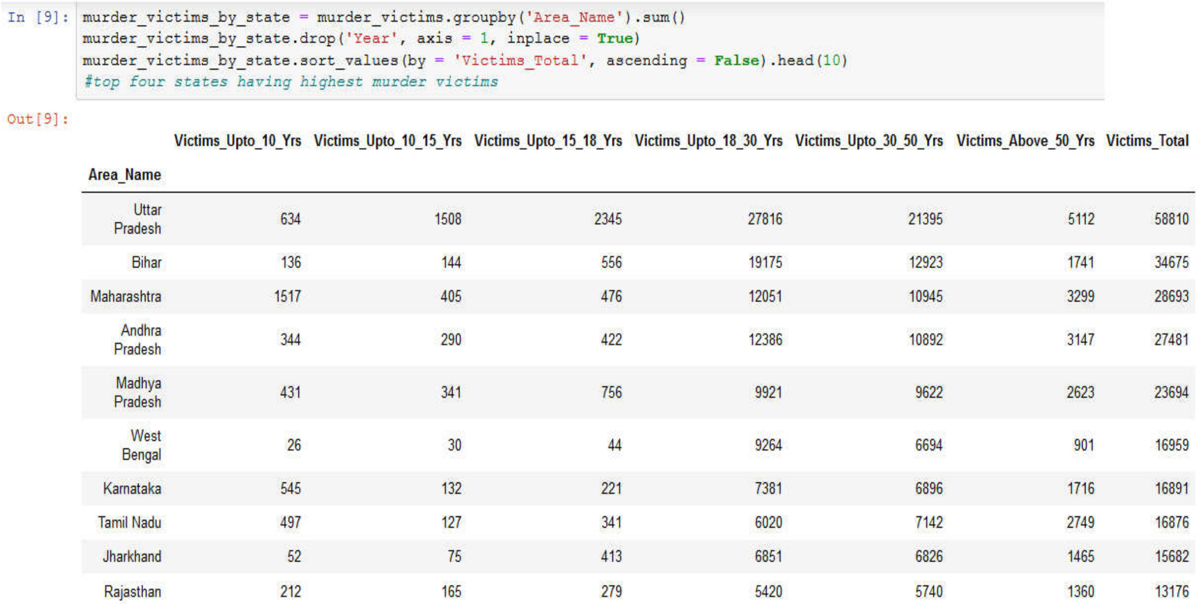


Fig 4: Male and female Murder victims

As per the Fig 4 visualization it has been observed that male victims are more compare to female victims.

2. States having higher murder victims percentage

Area_name and Victims_Total as input to identify states having higher murder victims percentage its corresponding data is given in Fig 5.



```
In [9]: murder_victims_by_state = murder_victims.groupby('Area_Name').sum()
        murder_victims_by_state.drop('Year', axis = 1, inplace = True)
        murder_victims_by_state.sort_values(by = 'Victims_Total', ascending = False).head(10)
        #top four states having highest murder victims
```

Out[9]:

	Victims_Upto_10_Yrs	Victims_Upto_10_15_Yrs	Victims_Upto_15_18_Yrs	Victims_Upto_18_30_Yrs	Victims_Upto_30_50_Yrs	Victims_Above_50_Yrs	Victims_Total
Area_Name							
Uttar Pradesh	634	1508	2345	27816	21395	5112	58810
Bihar	136	144	556	19175	12923	1741	34675
Maharashtra	1517	405	476	12051	10945	3299	28693
Andhra Pradesh	344	290	422	12386	10892	3147	27481
Madhya Pradesh	431	341	756	9921	9622	2623	23694
West Bengal	26	30	44	9264	6694	901	16959
Karnataka	545	132	221	7381	6896	1716	16891
Tamil Nadu	497	127	341	6020	7142	2749	16876
Jharkhand	52	75	413	6851	6826	1465	15682
Rajasthan	212	165	279	5420	5740	1360	13176

Fig 5: Displaying top 10 states having highest murder victims

Code:

murder_victims_by_state = murder_victims.groupby('Area_Name').sum()

murder_victims_by_state.drop('Year', axis = 1, inplace = True)

```
murder_victims_by_state.sort_values(by = 'Victims_Total', ascending = False).head(10)
```

3. Total murder victims in the years 2001 to 2010

Year,Area_Name and Victims_Total as input for calculation and visualizing total murder victims in the year 2001 to 2010

Code:

```
murder_victims_by_state=
murder_victims.groupby('Area_Name').sum()
murder_victims_by_state.drop('Year', axis = 1, inplace = True)

plt.subplots(figsize = (27, 10)
ct=murder_victims_by_state['Victims_Total'].sort_values(ascending = False)

ax = ct.plot.bar()
for p in ax.patches:
    ax.annotate(format(p.get_height()),
(p.get_x()+0.1,p.get_height()+1),fontsize=12,color='black')

ax.set_xlabel('Area Name')
ax.set_ylabel('Total Number of murder Victims from 2001 to 2010')
ax.set_title('Statewise total murder Victims throughout 2001 to 2010')
plt.show()
```

Fig 6 shows the visualization of Total murder victims in the years 2001 to 2010

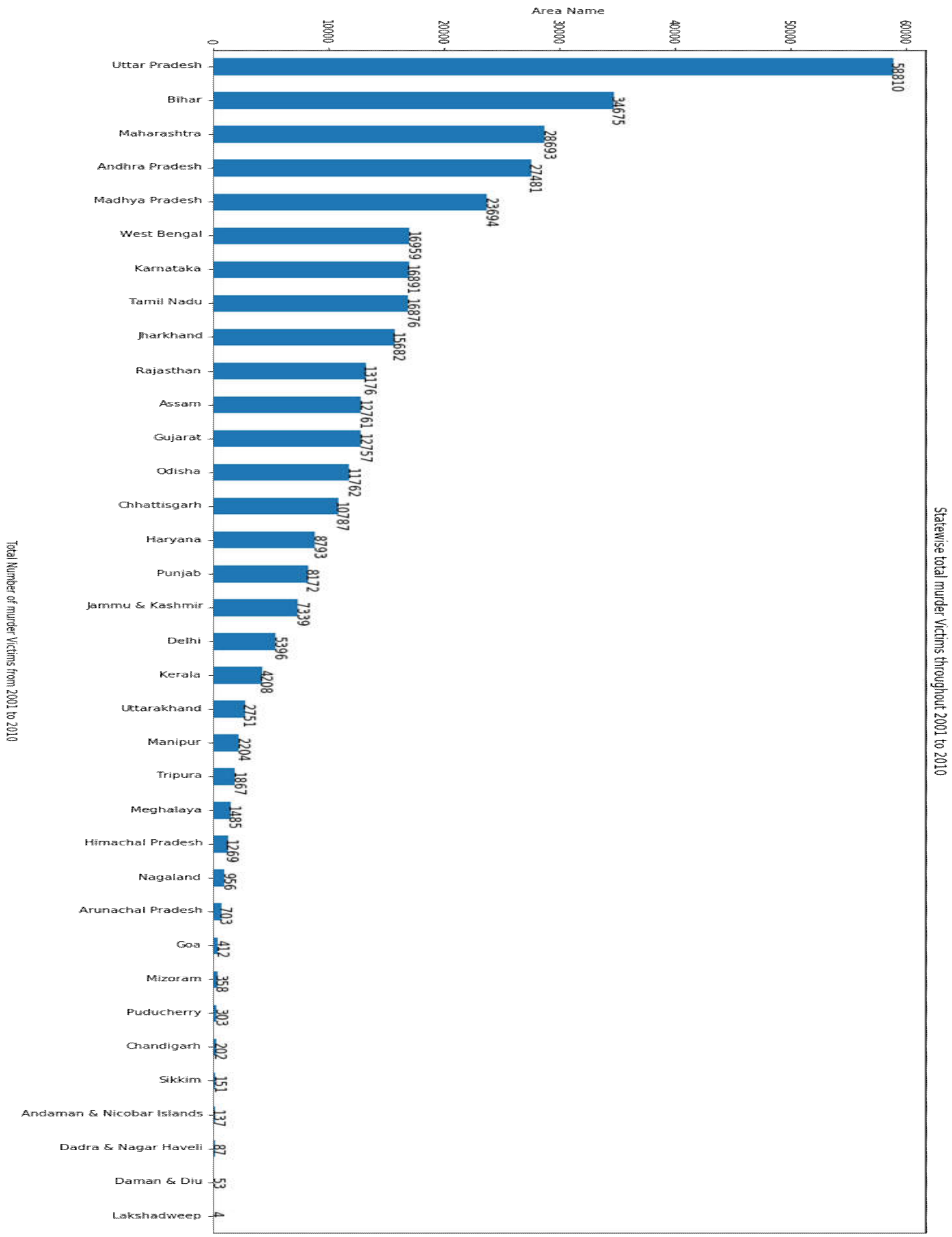


Fig 6: visualization of Total murder victims in the years 2001 to 2010

Fig 6 Displays all states with number of murder cases. Here uttar pradesh is having higher murder victims.On the X-axis Area_Name and On Y_axis total number of murder victims in the year 2001 to 2010.

4. Statewise murder victims based on Age group

Areaname,Victims_Above_50_Yrs,Victims_Tota,Victims_Upto_10_15_Yrs,Victims_Upto_10_ Yrs,Victims_Upto_15_18_Yrs,Victims_Upto_18_30_Yrs,Victims_Upto_30_50_Yrs considered for state wise murder victims based on age group.

Code:

```
murder_victims_heatmap=murder_victims_by_state.drop([ 'Victims_Total'], axis = 1)

plt.subplots(figsize = (10, 10))

ax = sns.heatmap(murder_victims_heatmap, cmap="Reds")

ax.set_xlabel('Age Group')

ax.set_ylabel('State Name')

ax.set_title('Statewise Victims of murder Cases based on Age Group')

plt.show()
```

We have observed state wise cases in each age group, here Uttar Pradesh is having higher murder percentage in the age group 18-30 yrs.

Fig 7 shows the visualization of the State wise murder victims based on Age group

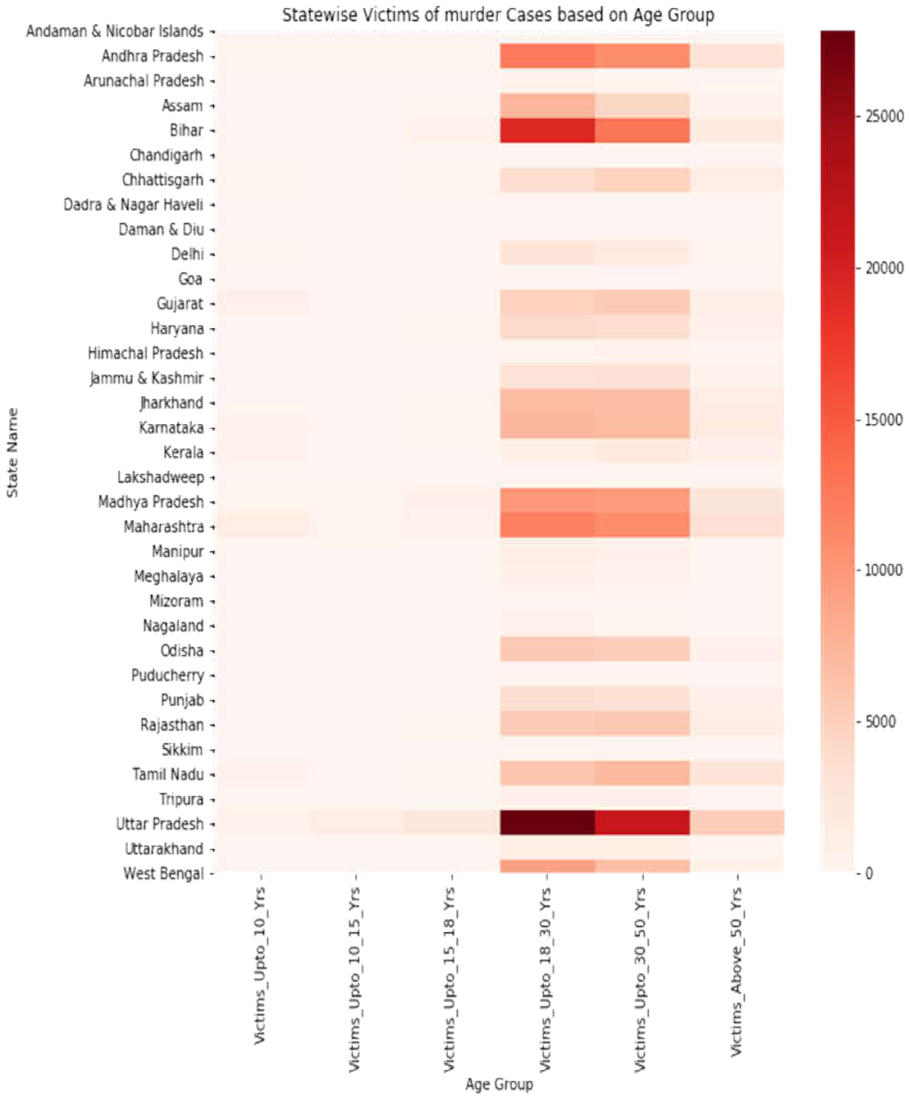


Fig 7: State wise murder victims based on Age group

5. Uttar Pradesh female murder victims percentage

Takes Female_Victims,Year,Victims_Total considered as input for the task.

Code:

```
up_murder_cases=up_murder_victims[up_murder_victims['Sub_Group_Name'] == '2. Female Victims']

plt.subplots(figsize = (15,6))

ct = up_murder_cases.groupby('Year').sum()
```

```
ax = ct['Victims_Total'].plot.bar()

for p in ax.patches:

   ax.annotate(format(p.get_height()), (p.get_x()+0.1, p.get_height()+2),fontsize=12)

ax.set_xlabel('Year')

ax.set_ylabel('Total Number of murder Victims from 2001 to 2010')

ax.set_title('Total murder Victims of Uttar Pradesh throught the Years 2001 to 2010')

plt.show()
```

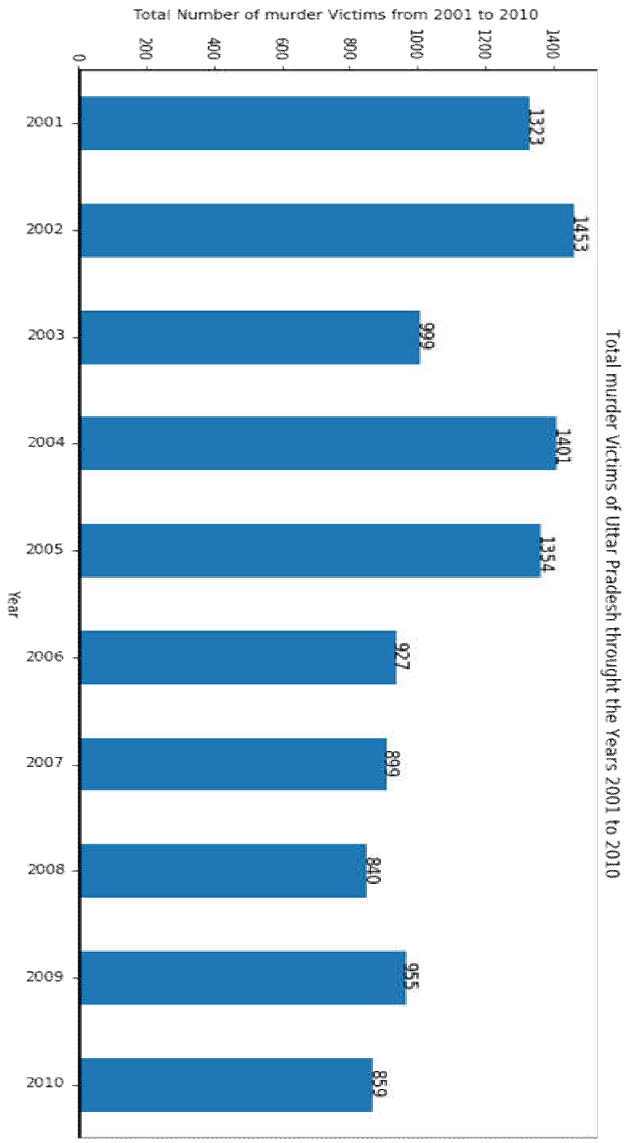


Fig 8 : Uttar Pradesh female murder victims percentage

6. Total number of murder victims in year 2002

Victims_Total, Area_Name, Year=2002 considered as input to identify total number of murder victims in year 2002.

Code:

```
import plotly.express as px

months1=murder_victims.query("Year==2002")

fig = px.pie(months1, values="Victims_Total", names="Area_Name",title="Total Number of
murders in 2002")

fig.show()
```

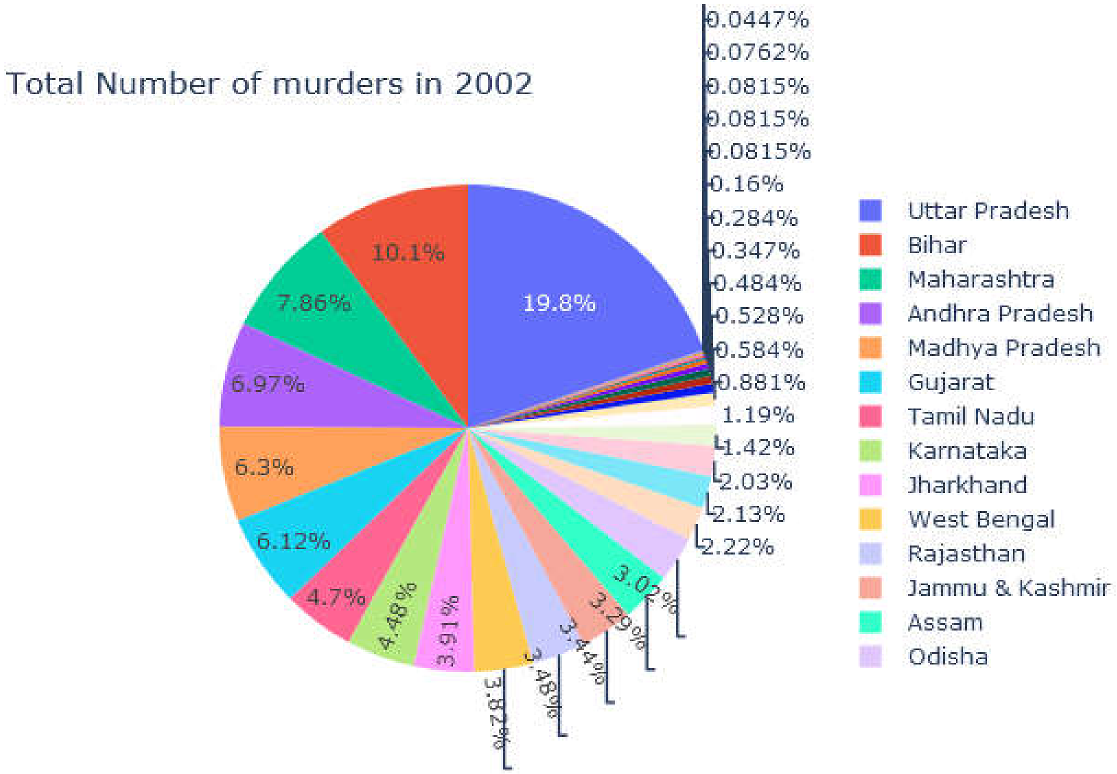


Fig 9: Total number of murder victims in year 2002

State wise murder victims in year 2002. Here Uttar Pradesh is having the higher murder victim percentage.

7. Murder victim’s growth in India.

Year and Victims_Total columns taken in consideration for identification of victim’s growth in India.

Code:

```
df=murder_victims.groupby(['Year']).sum()

df.loc[:,'Victims_Total'].plot(title='murder victims growth in India')
```

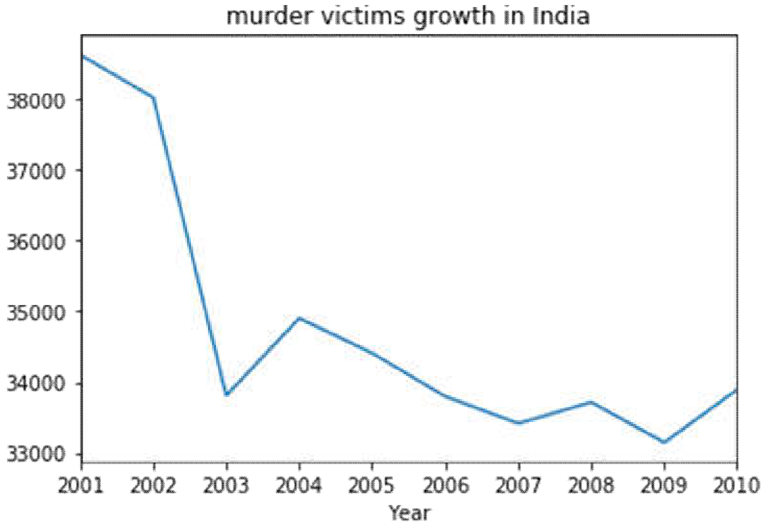


Fig 10: Murder victim’s growth in India

This graph shows the murder victims growth in India in 2001 we are having higher murder victims.

8. State wise murder victims of age 30 to 50 years

Victims_Total,Victims_Above_50_Yrs,Year,Area_Name considered as input

Code:

```
fig=px.scatter(murder_victims,x="Victims_Total",y="Victims_Upto_30_50_Yrs",color="Area_Name",title=" state wise murder victims above 50 years",log_x=True,size_max=55,range_x=[1,350], range_y=[1,200])

fig.show()
```

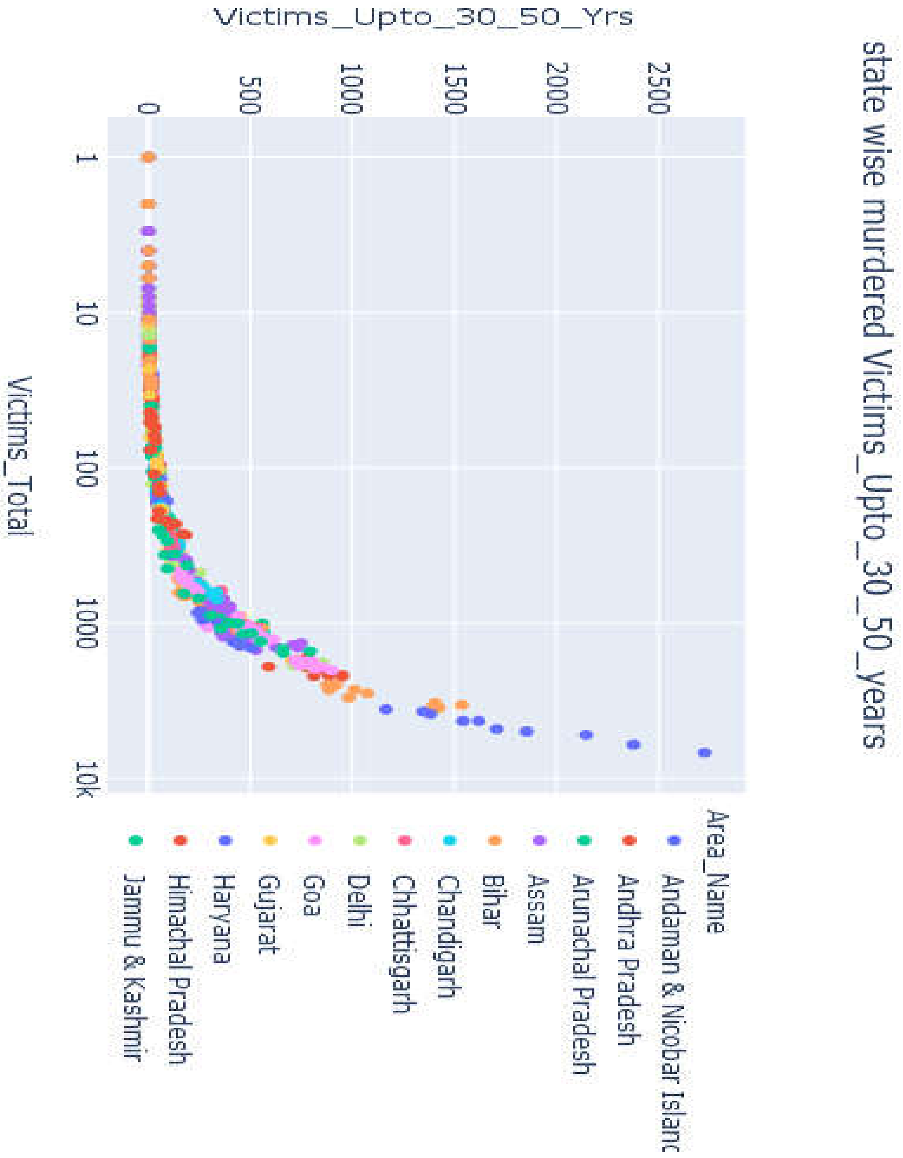


Fig11: State wise murder victims of Victims_Upto_30_50_ years

State wise murder victims between 30 to 50 years.We have taken this age because this age group is having more murder victims compared to other age groups .

9. Murder cases in Uttar Pradesh and Bihar year wise

Code:

```
b_murder_victims=murder_victims[murder_victims['Area_Name'] == 'Bihar']

# Let's have a look at yearly distribution of number of murder victims Bihar

b_murder_victims_by_year=b_murder_victims.groupby('Year').sum()

plt.subplots(figsize = (15, 6))

ax=(b_murder_victims_by_year['Victims_Total'].pct_change() * 100).plot(legend = True,label =
'Bihar')

(up_murder_victims_by_year['Victims_Total'].pct_change() * 100).plot(ax = ax, legend
=True,label = 'Uttar Pradesh')

ax.set(xlabel = 'Year', ylabel = 'Percent',

title = 'Yearly increase and decrease in number of murder cases in Bihar and Uttar Pradesh')

ax.axhline(0, color = 'black')

ax.axvline(2002, color = 'black')

plt.show()
```

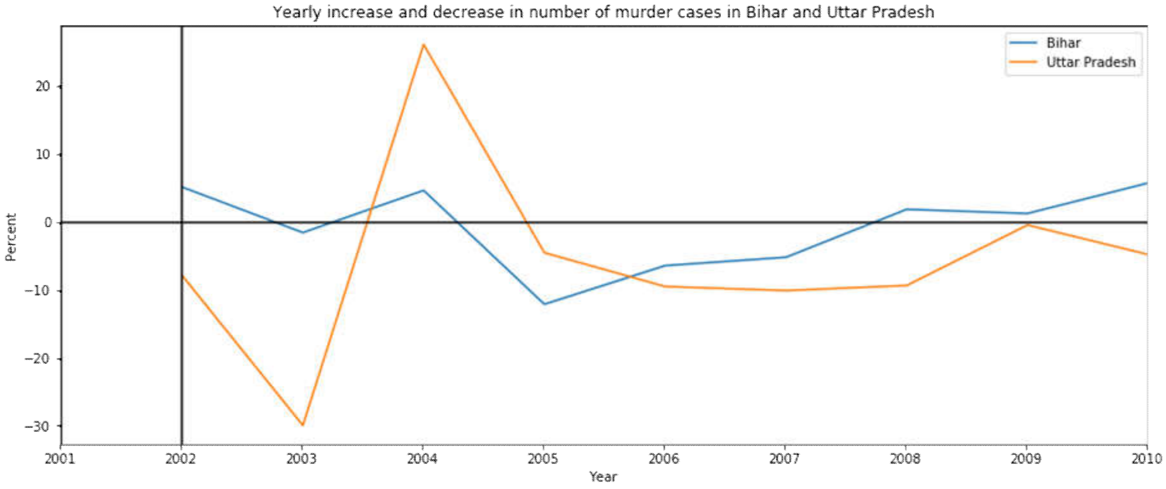


Fig 12 shows the Yearly increase and decrease in number of murder cases in Bihar and Uttar Pradesh

10. Minor and Major murder victims

Victims_Upto_10_Yrs,Victims_Upto_10_15_Yrs,
Victims_Upto_15_18_Yrs,Victims_Upto_18_30_Yrs, Victims_Upto_30_50_Yrs and
Victims_Above_50_Yrs considered as input for this task.

Code:

```
murder_minor=murder_victims['Victims_Upto_10_Yrs'].sum()+murder_victims['Victims_Upto_
10_15_Yrs'].sum()+murder_victims['Victims_Upto_15_18_Yrs'].sum()

murder_adults=murder_victims['Victims_Upto_18_30_Yrs'].sum()+murder_victims['Victims_U
pto_30_50_Yrs'].sum()+murder_victims['Victims_Above_50_Yrs'].sum()

data = [murder_minor,murder_adults ]

index =['Major Victims', 'Minor Victims']

data_df = pd.DataFrame(data, columns = ['number of victims'], index = index)

#print(data)

plt.figure(figsize=(4,3), edgecolor='black')

ax = data_df.plot.bar(rot=0)

for p in ax.patches:

   ax.annotate(format(p.get_height()), (p.get_x()+0.1, p.get_height()+3),fontsize=12)

plt.title("Minor vs Major -Victims",fontsize=14)
```

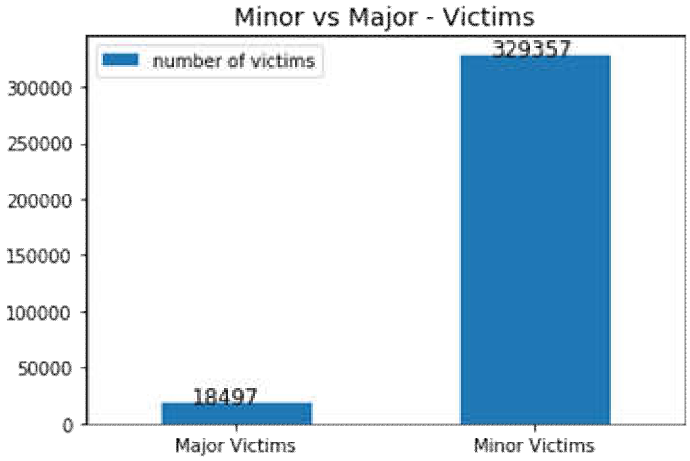


Fig 13: Minor and Major murder victims

Conclusion

Taking male and female murder victims as an input here we can get to know that male murder victims are more compared to female murder victims. The to 5 sates having more murder victims are Uttar Pradesh, Bihar, Maharashtra, Andhra Pradesh, Madhya Pradesh. The total 35 states murder victims here the Uttar Pradesh is having 58810 murder victims and lowest is Lakshadweep is having 4.Based on different age groups the murder cases are represented here. The Uttar Pradesh is having the more murder cases. Here we have taken Uttar Pradesh state because it is having highest murder victims when compared to other states. In specific year 2002 the murder victims are represented in pie plot. The murder growth in total states is represented in line plot. Statewide total murders in bar graph using different colored plots for different representation of states. Age group in between 30-50 yrs the murder victims are represented using scatter plot. The yearly increase and decrease in number of murder cases of 2 states Bihar and Uttar Pradesh are represented. The minor and major victims are represented the minor age groups are taken as Victims_Upto_10_Yrs, Victims_Upto_10_15_Yrs, Victims_Upto_15_18_Yrs and the major are taken as Victims_Upto_18_30_Yrs,Victims_Upto_30_50_Yrs, Victims_Above_50_Yrs.

References

1. https://en.wikipedia.org/wiki/Category:Indian_murder_victims
2. https://www.python.org/
3. van der Walt, Stéfan & Colbert, S. & Varoquaux, Gael. (2011). The NumPy Array: A Structure for Efficient Numerical Computation. Computing in Science & Engineering. 13. 22 - 30. 10.1109/MCSE.2011.37
4. Barrett, Paul & Hunter, J. & Miller, J.T. & Hsu, J.-C & Greenfield, P.. (2005). matplotlib -- A Portable Python Plotting Package.
5. https://plotly.com/
6. https://packaging.python.org/tutorials/installing-packages/
7. https://statinfer.com/104-2-2-practice-working-with-datasets-in-python/
8. https://www.shanelynn.ie/python-pandas-read_csv-load-data-from-csv-files/
9. https://jakevdp.github.io/PythonDataScienceHandbook/03.03-operations-in-pandas.html
10. https://www.datacamp.com/community/tutorials/matplotlib-tutorial-python