Project 1: Analysis of Police-Inflicted Fatalities

Darren Pardo
Henry Sanchez
Jakub Tamulewicz
Joshua Marcus

Abstract

- Goal: uncover patterns related to US police-inflicted fatalities
- Clean data for useful and relevant information
- Combine with US Census population data to investigate trends related to fatalities

Sources

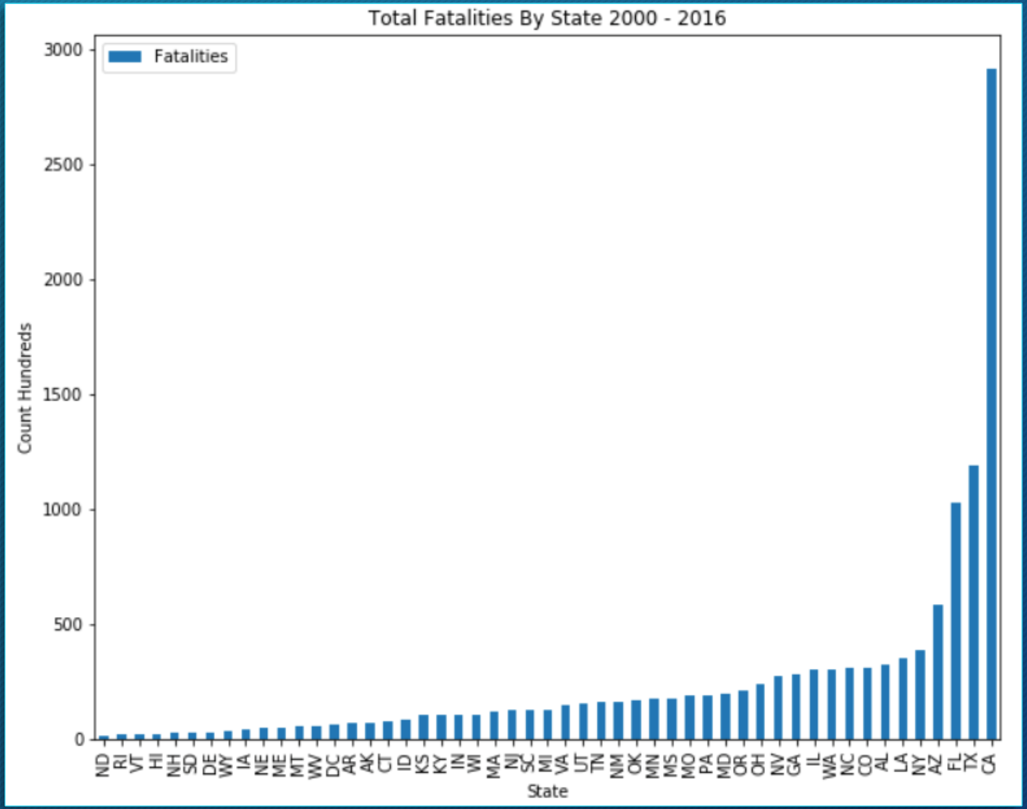
- Primary source: Individuals Killed by Police in the US from 2000 – 2016
  - CSV of people killed by police over a 16 year span in the US
  - Victim data includes: age, gender, location, manner of death, etc.
  - https://data.world/awram/us-police-involved-fatalities
- Secondary source: United States Census Bureau
  - Used to investigate trends between fatalities and population change
  - https://www.census.gov/

Questions

1. What trends can we uncover regarding the frequency of police-involved fatalities from 2000 to 2016?
   a. What state has the highest recorded number of fatalities?
   b. What is the most common manner of death?
   c. Does the month of the year influence the number of fatalities?
   d. Which gender is the most affected?
   e. What age group is most affected?
2. Combining census info with our dataset, can we show a direct correlation between increase in population and increase in fatalities?

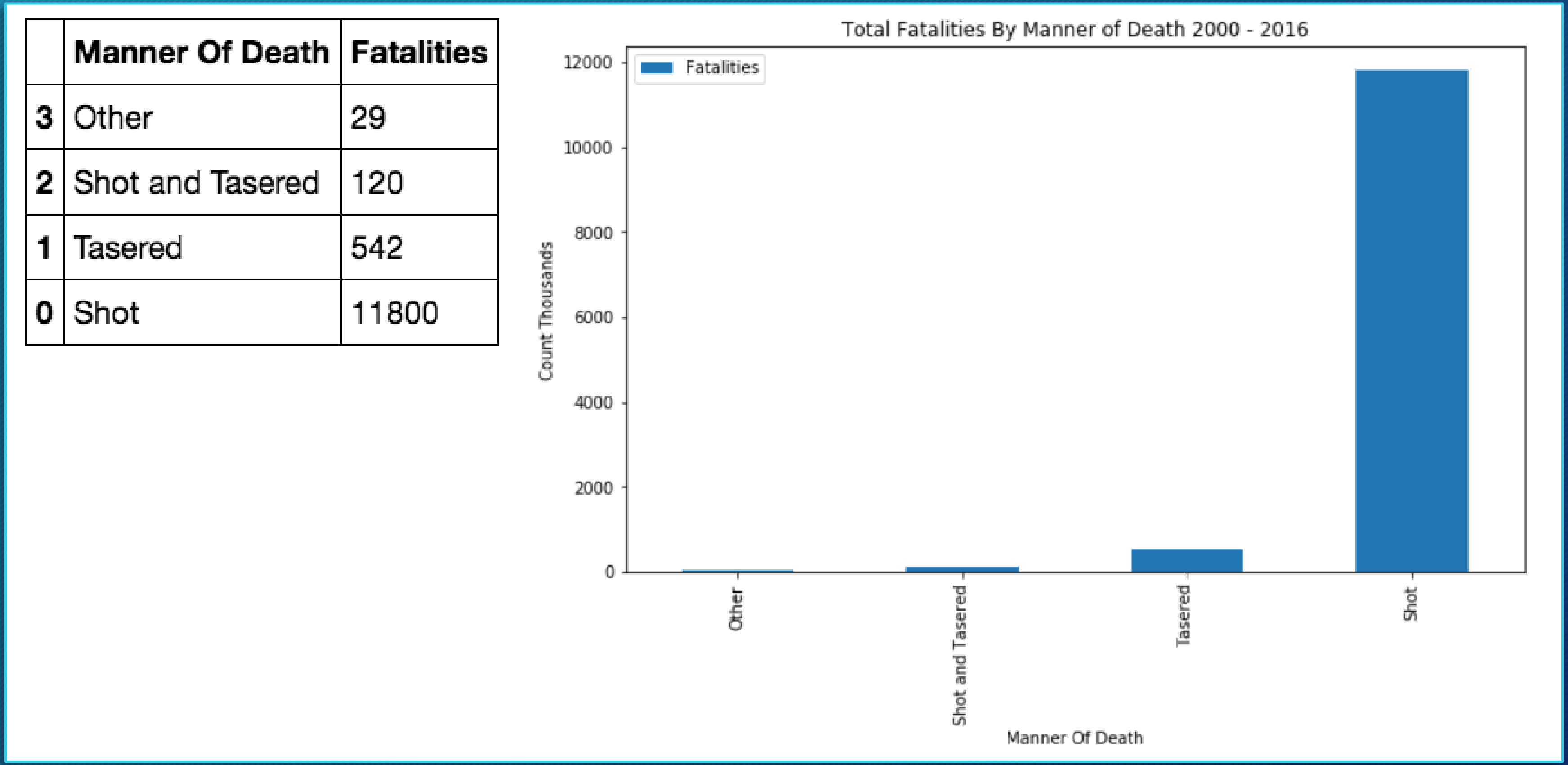
1. We Discovered the Following Trends:

a. The state with the highest recorded police-inflicted fatalities over the 16 year span is California.



1. We Discovered the Following Trends:

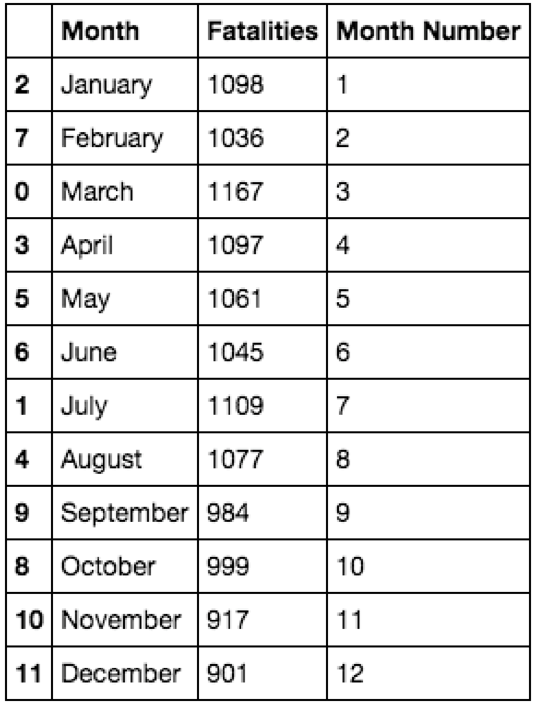
b. The most common manner of death in these fatalities is by firearm.



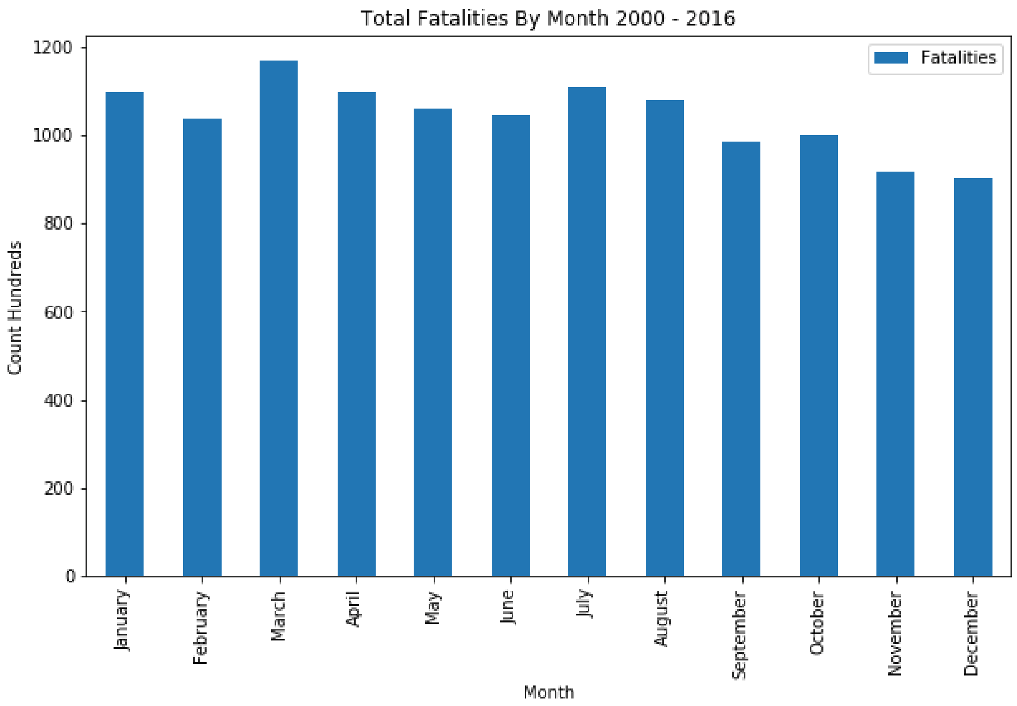
	Manner Of Death	Fatalities
3	Other	29
2	Shot and Tasered	120
1	Tasered	542
0	Shot	11800

1. We Discovered the Following Trends:

c. The month with the overall lowest police fatalities was December, with 901 fatalities. The highest was March, with 1167. Our dataset shows a marginal spike for March, and a general decrease throughout the subsequent months.

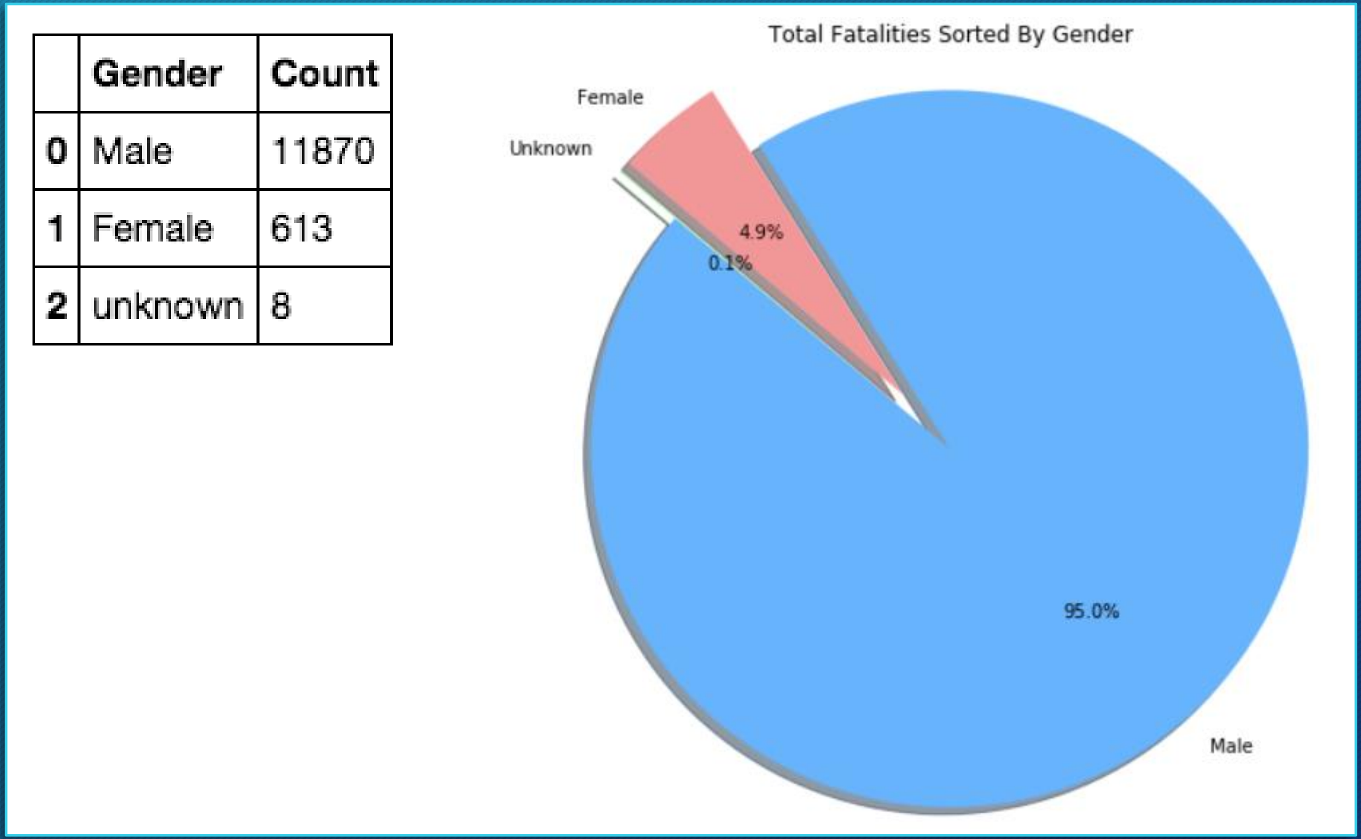


	Month	Fatalities	Month Number
2	January	1098	1
7	February	1036	2
0	March	1167	3
3	April	1097	4
5	May	1061	5
6	June	1045	6
1	July	1109	7
4	August	1077	8
9	September	984	9
8	October	999	10
10	November	917	11
11	December	901	12



1. We Discovered the Following Trends:

d. The most affected gender, by a significant margin, is male.

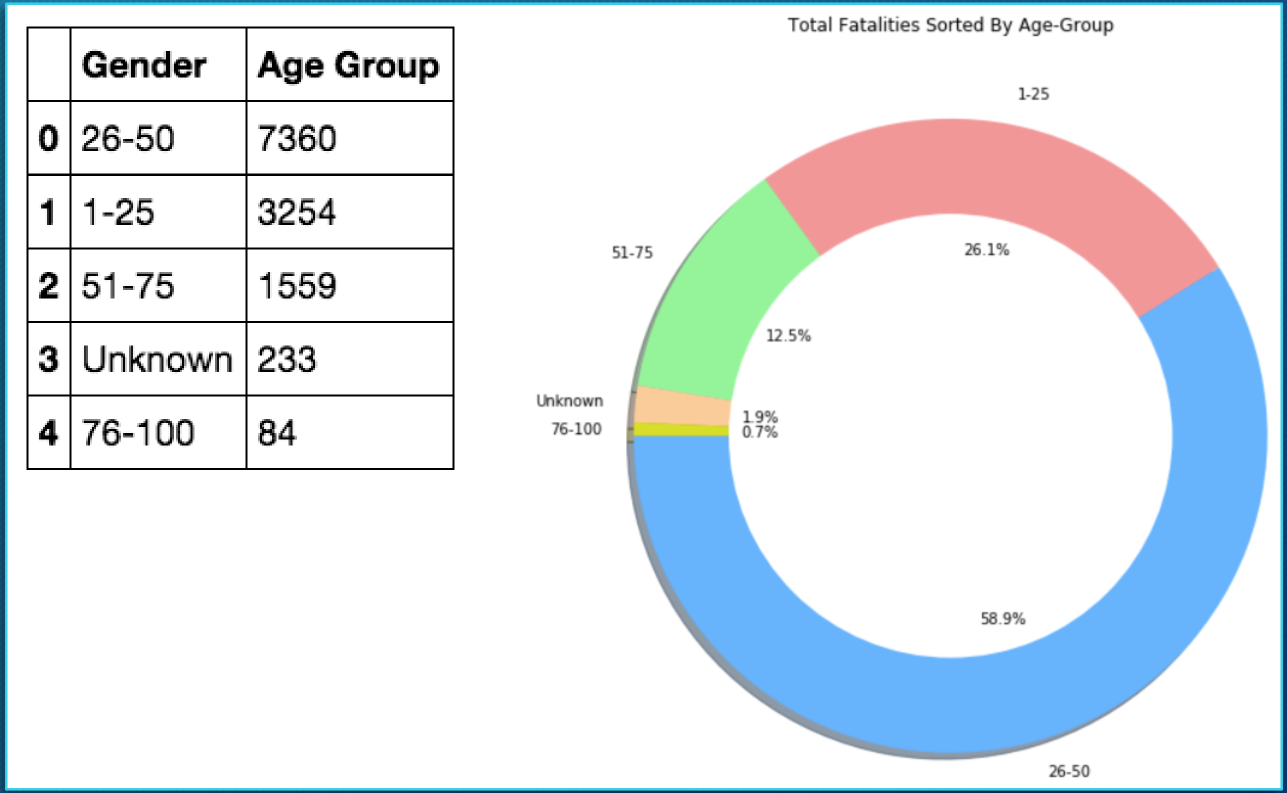


	Gender	Count
0	Male	11870
1	Female	613
2	unknown	8

Total Fatalities Sorted By Gender

1. We Discovered the Following Trends:

e. The most affected age group is 25 to 50 year-olds. The second highest group was 25 years and younger.

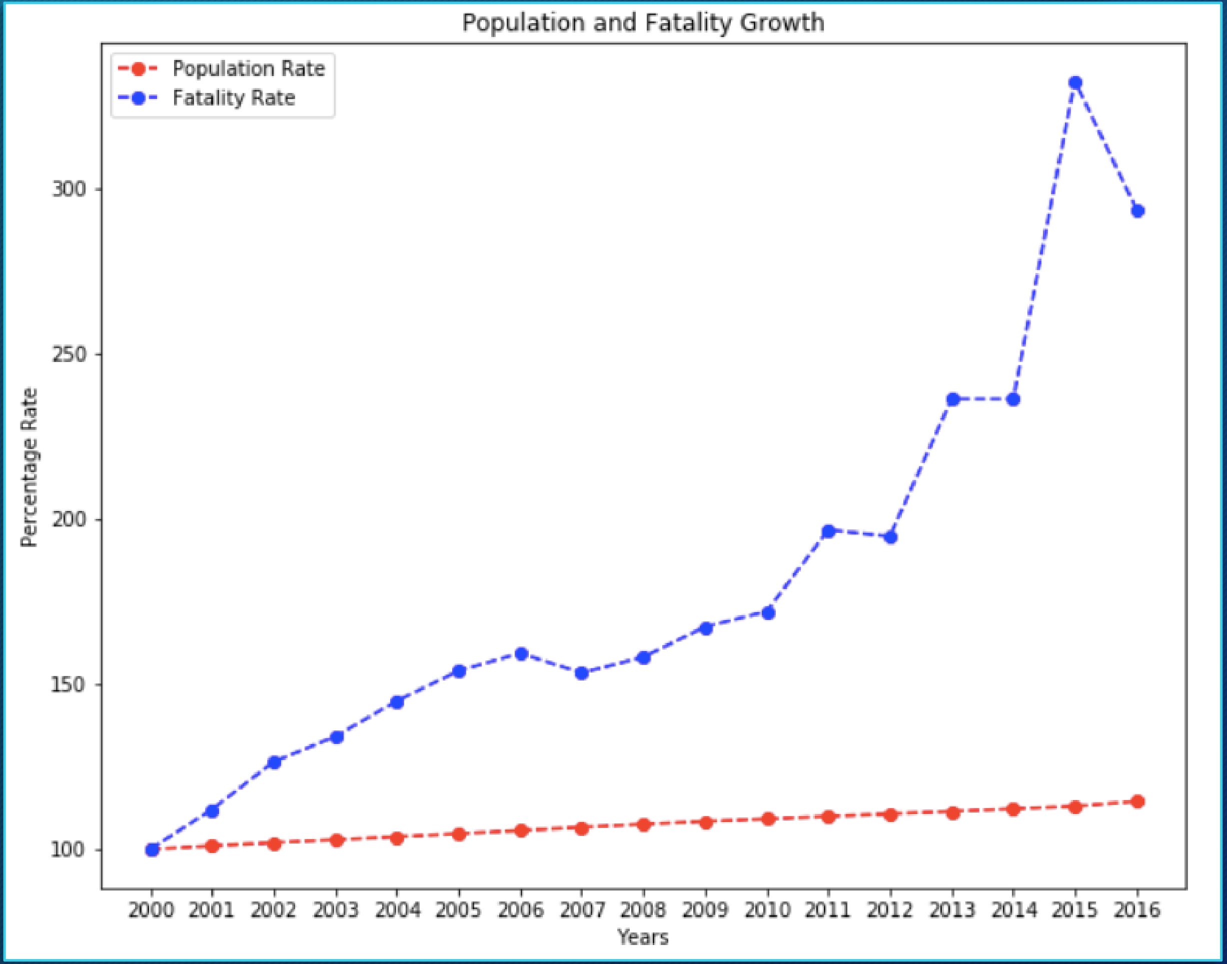


	Gender	Age Group
0	26-50	7360
1	1-25	3254
2	51-75	1559
3	Unknown	233
4	76-100	84

Total Fatalities Sorted By Age-Group

2. Fatality Rate vs. US Population Growth

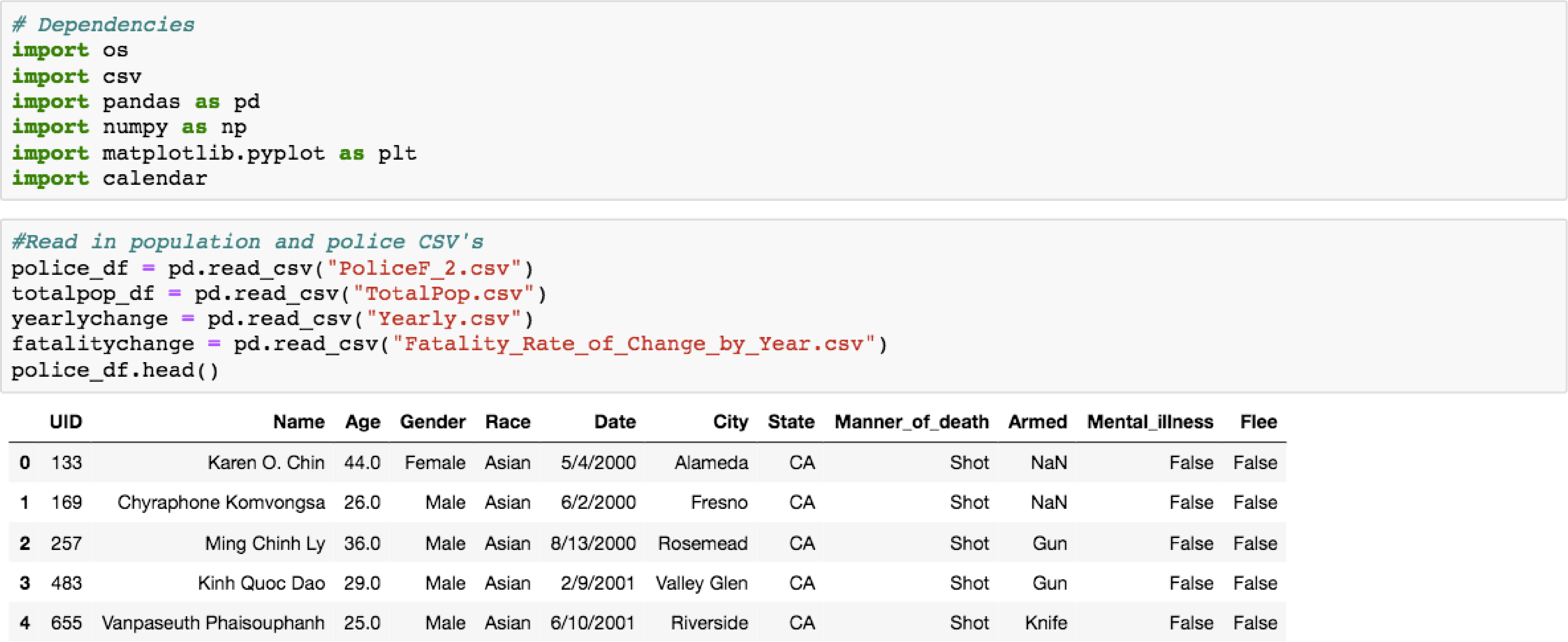
Recorded police fatalities have, overall, grown at an alarmingly higher rate, as compared to overall US Population growth



Coding Workflow

- Planning: Excel and Pivot Tables
    - Opening our primary data source (Police Fatalities CSV file) in Excel
    - Planning how to clean CSV files in a formulaic manner
    - Creating pivot tables to visualize the data quickly and to get a sense of trends
- Jupyter Notebook
    - Read in CSVs
    - Convert CSVs into data frames and clean data
    - Plot graphs from data in data frames

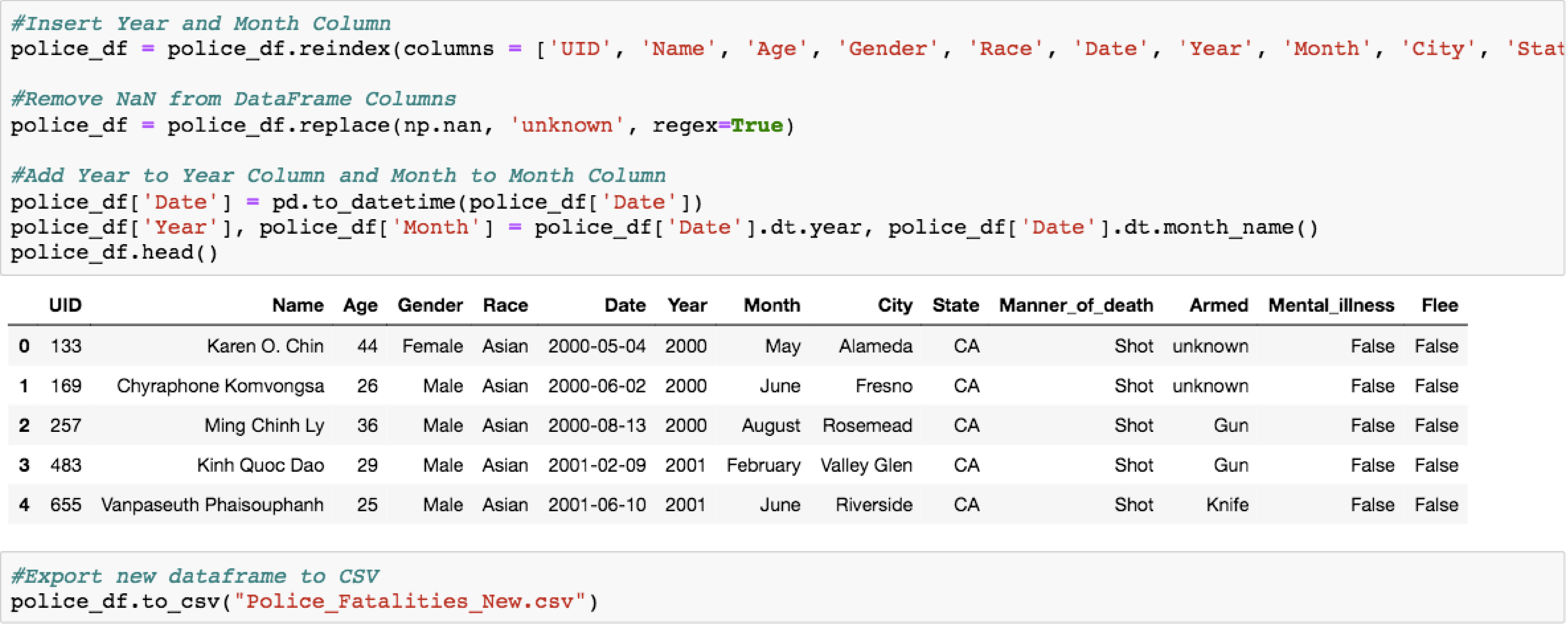
Jupyter Notebook Code Breakdown



```python
# Dependencies
import os
import csv
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import calendar
```

```python
#Read in population and police CSV's
police_df = pd.read_csv("PoliceF_2.csv")
totalpop_df = pd.read_csv("TotalPop.csv")
yearlychange = pd.read_csv("Yearly.csv")
fatalitychange = pd.read_csv("Fatality_Rate_of_Change_by_Year.csv")
police_df.head()
```

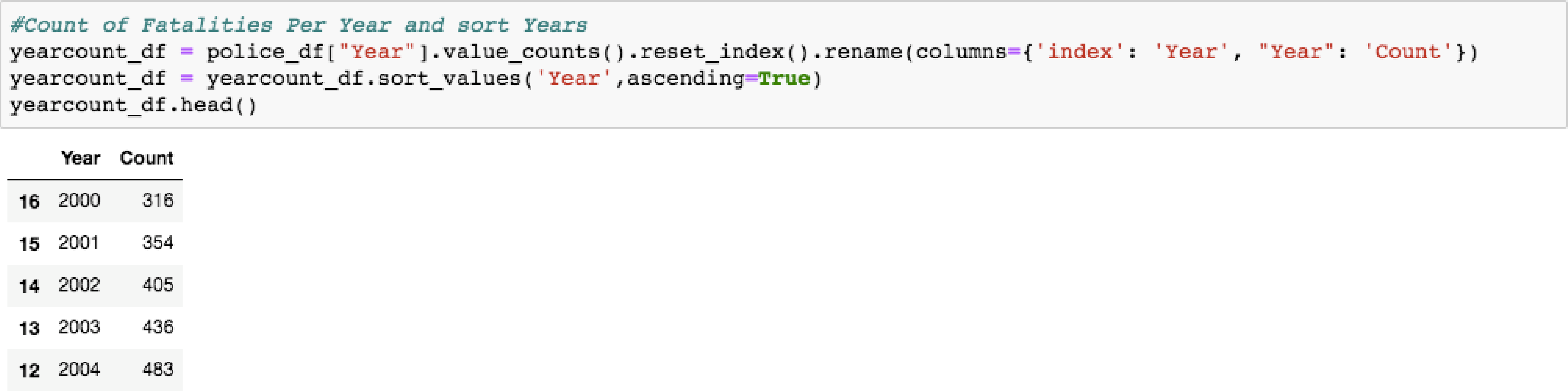
	UID	Name	Age	Gender	Race	Date	City	State	Manner_of_death	Armed	Mental_illness	Flee
0	133	Karen O. Chin	44.0	Female	Asian	5/4/2000	Alameda	CA	Shot	NaN	False	False
1	169	Chyraphone Komvongsa	26.0	Male	Asian	6/2/2000	Fresno	CA	Shot	NaN	False	False
2	257	Ming Chinh Ly	36.0	Male	Asian	8/13/2000	Rosemead	CA	Shot	Gun	False	False
3	483	Kinh Quoc Dao	29.0	Male	Asian	2/9/2001	Valley Glen	CA	Shot	Gun	False	False
4	655	Vanpaseuth Phaisouphanh	25.0	Male	Asian	6/10/2001	Riverside	CA	Shot	Knife	False	False



```python
#Insert Year and Month Column
police_df = police_df.reindex(columns = ['UID', 'Name', 'Age', 'Gender', 'Race', 'Date', 'Year', 'Month', 'City', 'Stat

#Remove NaN from DataFrame Columns
police_df = police_df.replace(np.nan, 'unknown', regex=True)

#Add Year to Year Column and Month to Month Column
police_df['Date'] = pd.to_datetime(police_df['Date'])
police_df['Year'], police_df['Month'] = police_df['Date'].dt.year, police_df['Date'].dt.month_name()
police_df.head()
```

	UID	Name	Age	Gender	Race	Date	Year	Month	City	State	Manner_of_death	Armed	Mental_illness	Flee
0	133	Karen O. Chin	44	Female	Asian	2000-05-04	2000	May	Alameda	CA	Shot	unknown	False	False
1	169	Chyraphone Komvongsa	26	Male	Asian	2000-06-02	2000	June	Fresno	CA	Shot	unknown	False	False
2	257	Ming Chinh Ly	36	Male	Asian	2000-08-13	2000	August	Rosemead	CA	Shot	Gun	False	False
3	483	Kinh Quoc Dao	29	Male	Asian	2001-02-09	2001	February	Valley Glen	CA	Shot	Gun	False	False
4	655	Vanpaseuth Phaisouphanh	25	Male	Asian	2001-06-10	2001	June	Riverside	CA	Shot	Knife	False	False

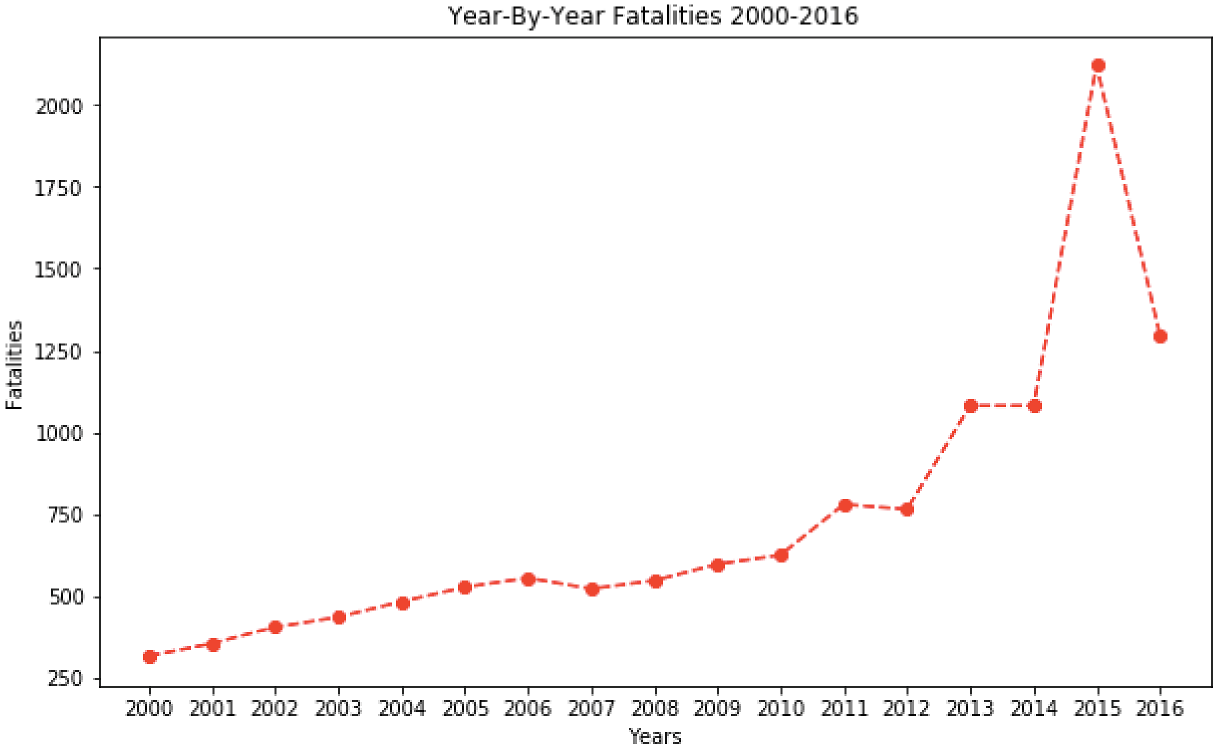
```python
#Export new dataframe to CSV
police_df.to_csv("Police_Fatalities_New.csv")
```



```python
#Count of Fatalities Per Year and sort Years
yearcount_df = police_df["Year"].value_counts().reset_index().rename(columns={'index': 'Year', "Year": 'Count'})
yearcount_df = yearcount_df.sort_values('Year',ascending=True)
yearcount_df.head()
```

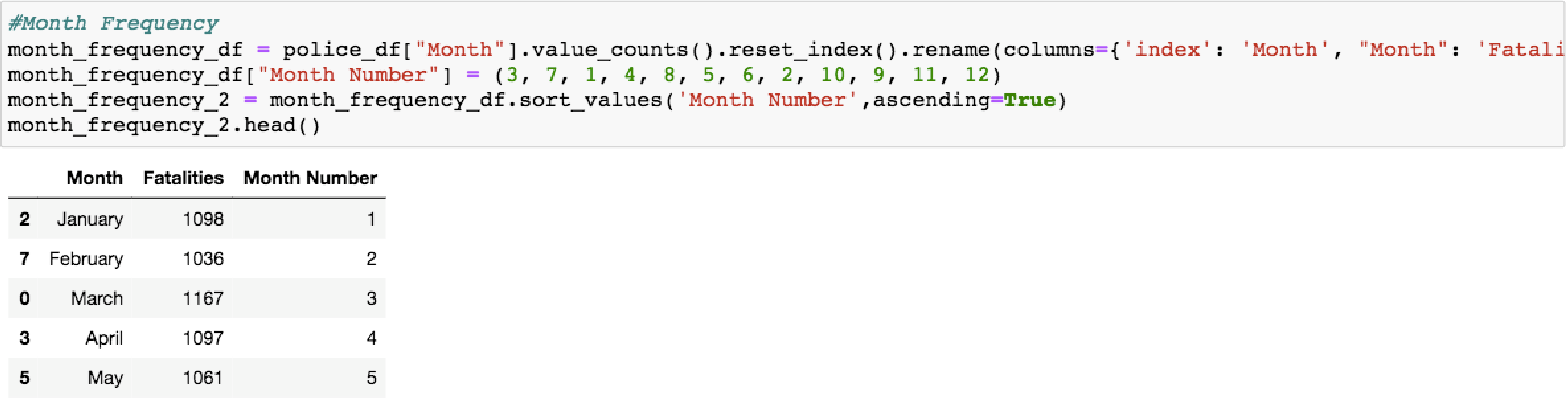
```
    Year  Count
16  2000    316
15  2001    354
14  2002    405
13  2003    436
12  2004    483
```

```
#Count of Fatalities Per Year
plt.plot(yearcount_df.Year, yearcount_df.Count, linestyle='--', marker='o', color="r")
plt.rcParams['figure.figsize'] = (8,5)
plt.title("Year-By-Year Fatalities 2000-2016")
plt.ylabel('Fatalities')
plt.xlabel('Years')
plt.xticks([2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016])
plt.show()
fig3 = plt.gcf()
```



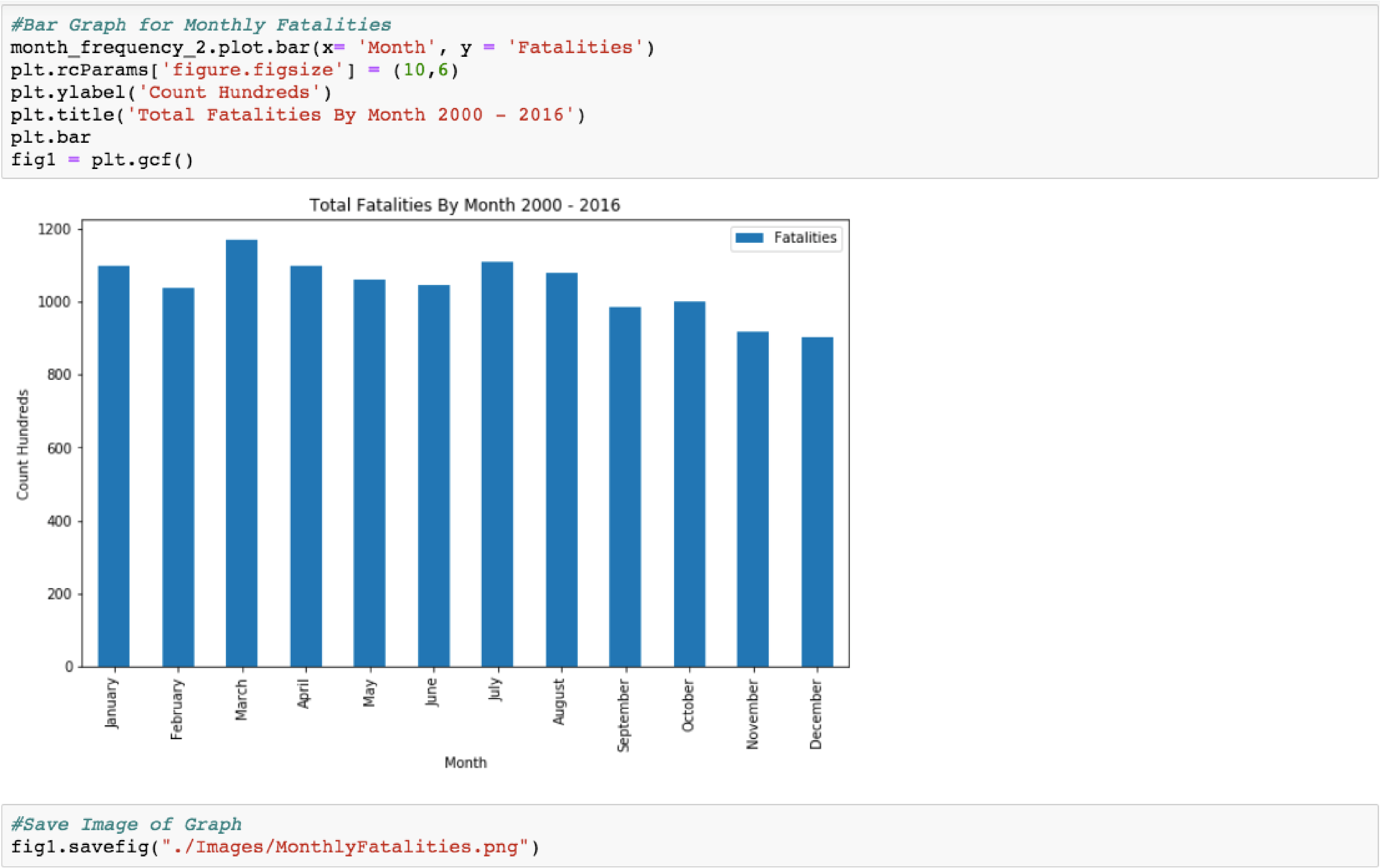
```
<Figure size 576x360 with 0 Axes>
```

```
#Save Image of Graph
fig3.savefig("./Images/FatYear.png")
```



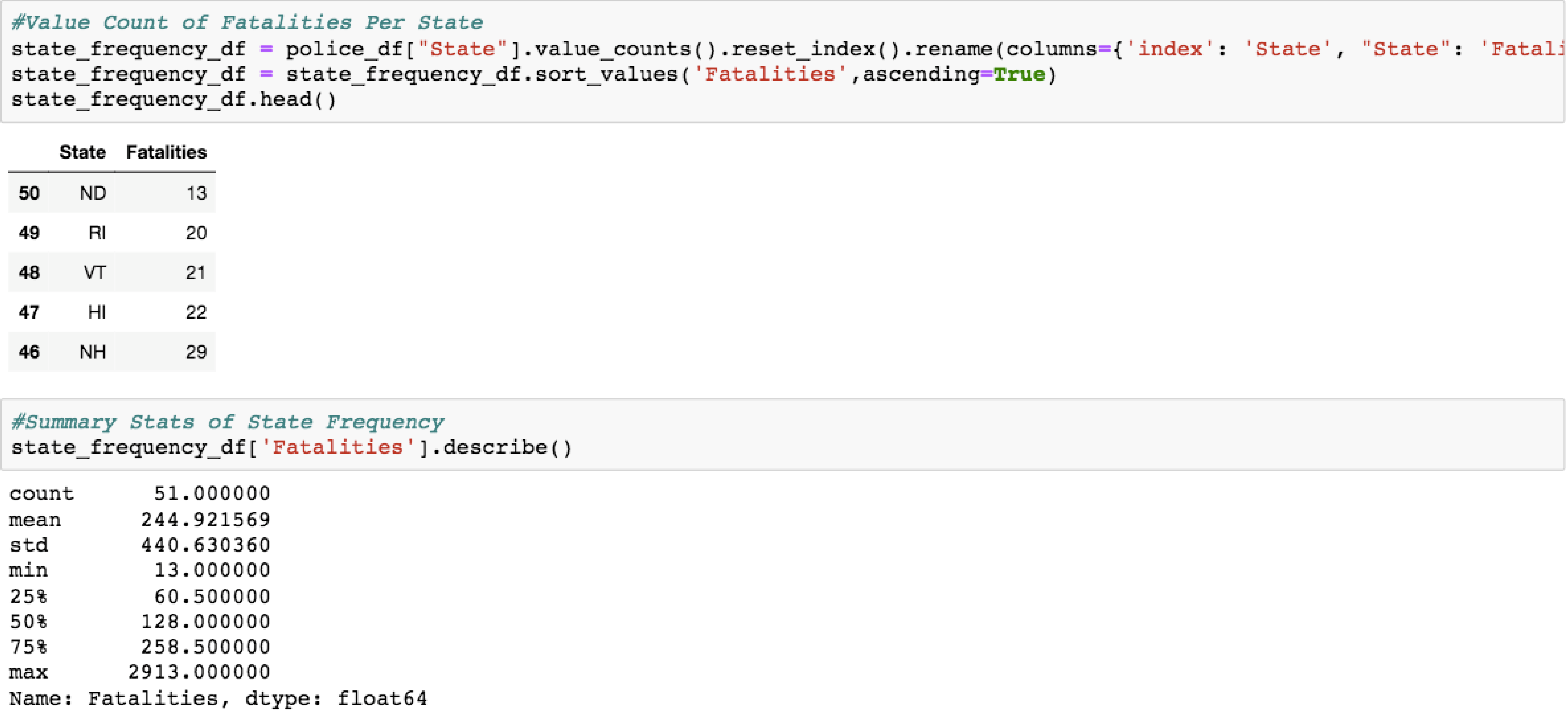
```python
#Month Frequency
month_frequency_df = police_df["Month"].value_counts().reset_index().rename(columns={'index': 'Month', "Month": 'Fatali
month_frequency_df["Month Number"] = (3, 7, 1, 4, 8, 5, 6, 2, 10, 9, 11, 12)
month_frequency_2 = month_frequency_df.sort_values('Month Number',ascending=True)
month_frequency_2.head()
```

	Month	Fatalities	Month Number
2	January	1098	1
7	February	1036	2
0	March	1167	3
3	April	1097	4
5	May	1061	5



```
#Bar Graph for Monthly Fatalities
month_frequency_2.plot.bar(x= 'Month', y = 'Fatalities')
plt.rcParams['figure.figsize'] = (10,6)
plt.ylabel('Count Hundreds')
plt.title('Total Fatalities By Month 2000 - 2016')
plt.bar
fig1 = plt.gcf()
```

```
#Save Image of Graph
fig1.savefig("./Images/MonthlyFatalities.png")
```

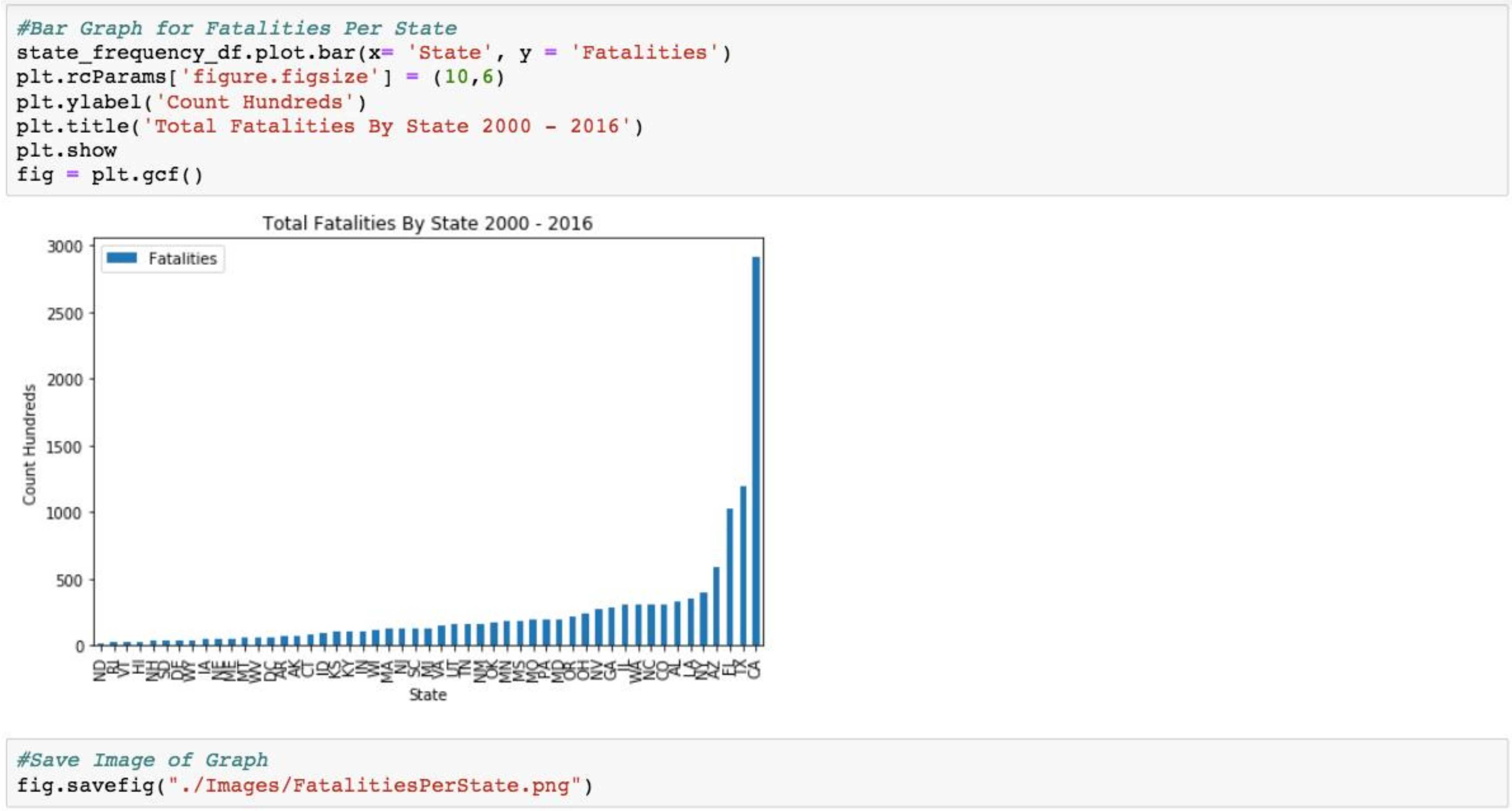


```python
#Value Count of Fatalities Per State
state_frequency_df = police_df["State"].value_counts().reset_index().rename(columns={'index': 'State', "State": 'Fatali
state_frequency_df = state_frequency_df.sort_values('Fatalities',ascending=True)
state_frequency_df.head()
```

	State	Fatalities
50	ND	13
49	RI	20
48	VT	21
47	HI	22
46	NH	29

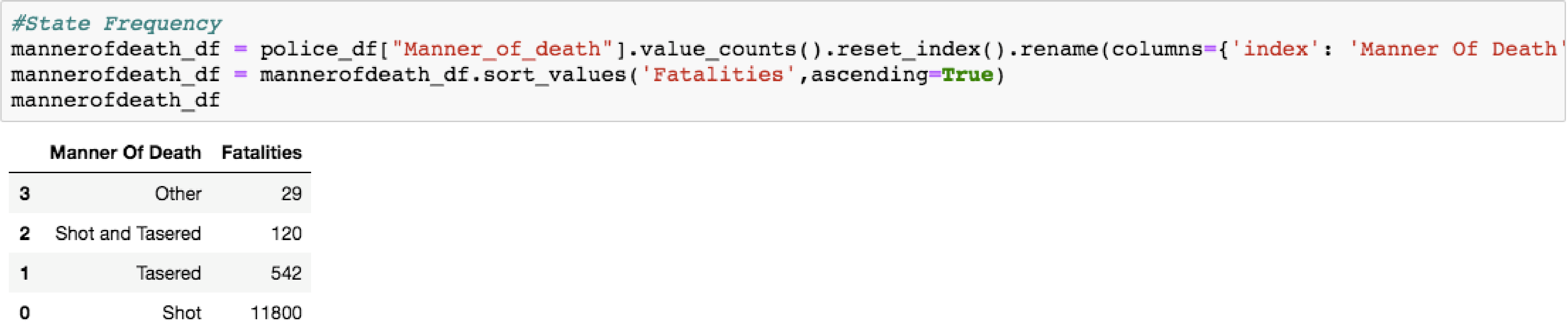
```python
#Summary Stats of State Frequency
state_frequency_df['Fatalities'].describe()
```

```
count      51.000000
mean      244.921569
std       440.630360
min        13.000000
25%        60.500000
50%       128.000000
75%       258.500000
max      2913.000000
Name: Fatalities, dtype: float64
```



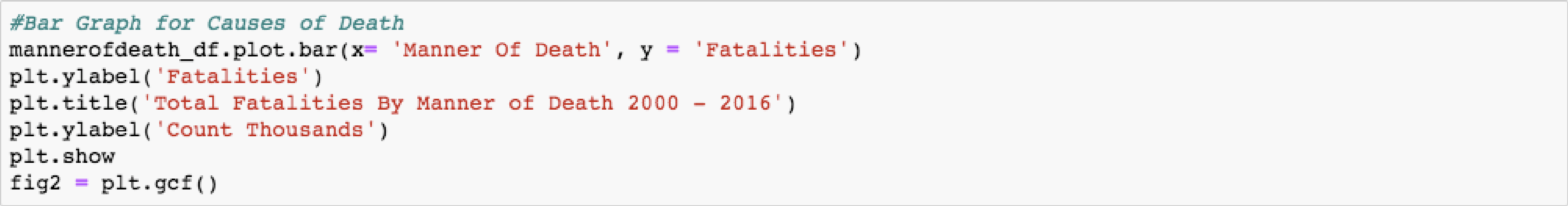
```
#Bar Graph for Fatalities Per State
state_frequency_df.plot.bar(x= 'State', y = 'Fatalities')
plt.rcParams['figure.figsize'] = (10,6)
plt.ylabel('Count Hundreds')
plt.title('Total Fatalities By State 2000 - 2016')
plt.show
fig = plt.gcf()
```

```
#Save Image of Graph
fig.savefig("./Images/FatalitiesPerState.png")
```

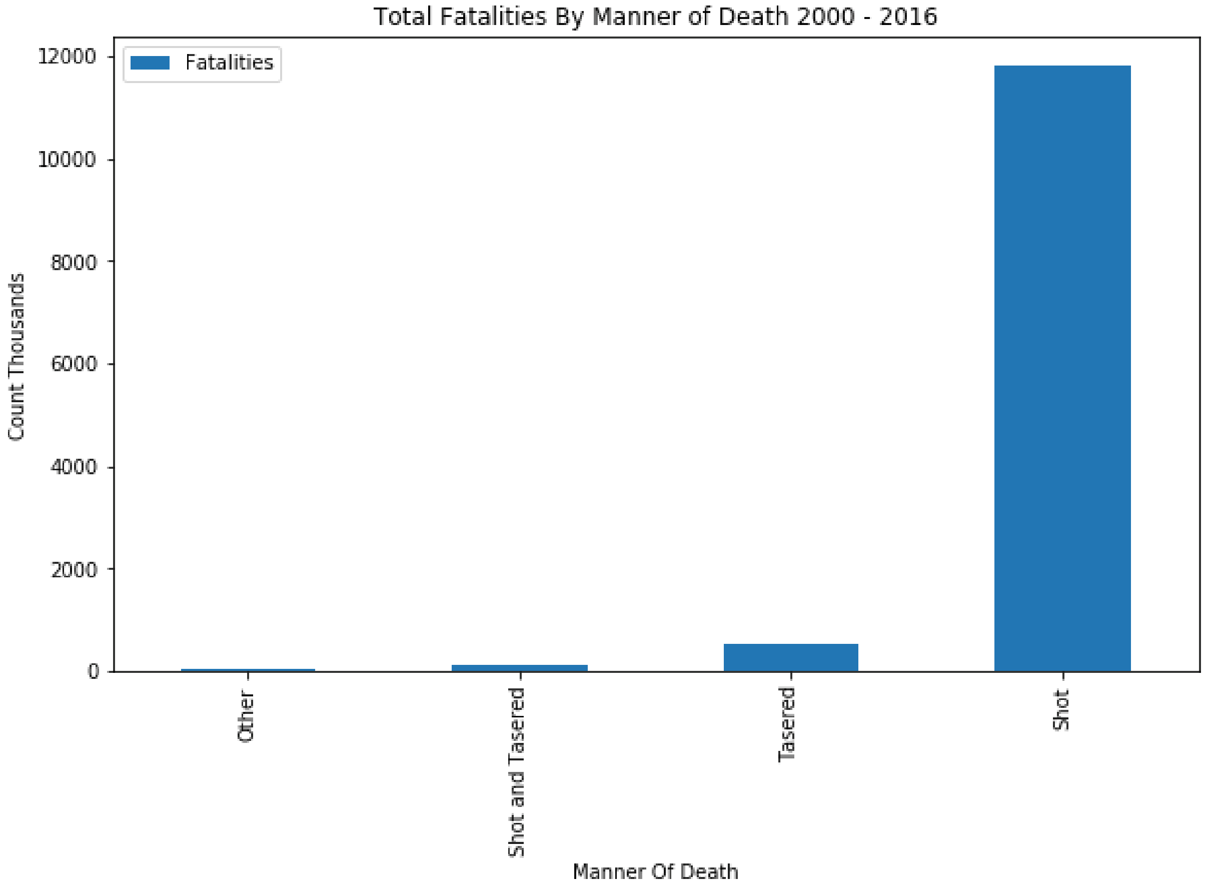


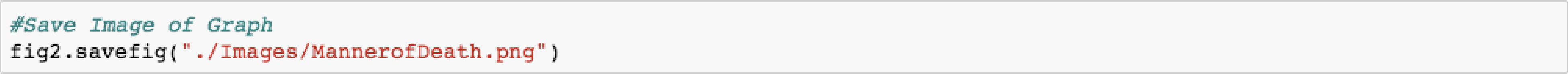
```
#State Frequency
mannerofdeath_df = police_df["Manner_of_death"].value_counts().reset_index().rename(columns={'index': 'Manner Of Death'
mannerofdeath_df = mannerofdeath_df.sort_values('Fatalities',ascending=True)
mannerofdeath_df
```

```
      Manner Of Death  Fatalities
3               Other          29
2    Shot and Tasered         120
1             Tasered         542
0                Shot       11800
```

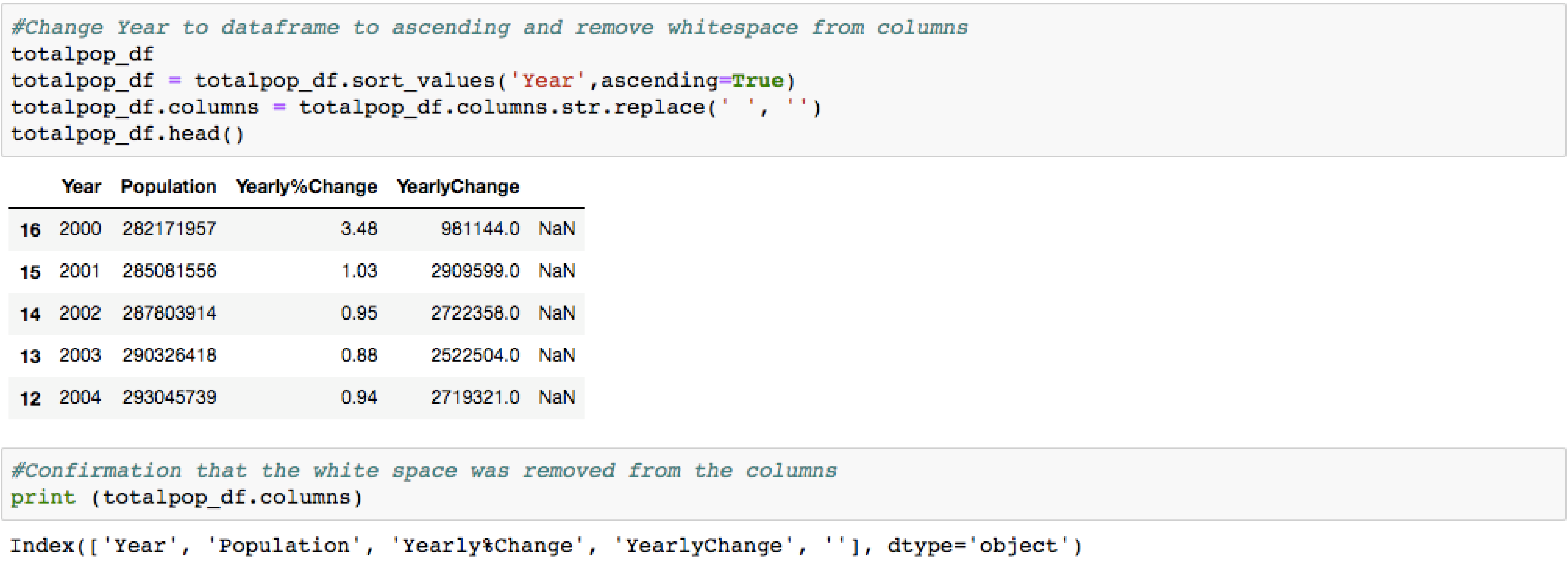


```
#Bar Graph for Causes of Death
mannerofdeath_df.plot.bar(x= 'Manner Of Death', y = 'Fatalities')
plt.ylabel('Fatalities')
plt.title('Total Fatalities By Manner of Death 2000 - 2016')
plt.ylabel('Count Thousands')
plt.show
fig2 = plt.gcf()
```





```
#Save Image of Graph
fig2.savefig("./Images/MannerofDeath.png")
```

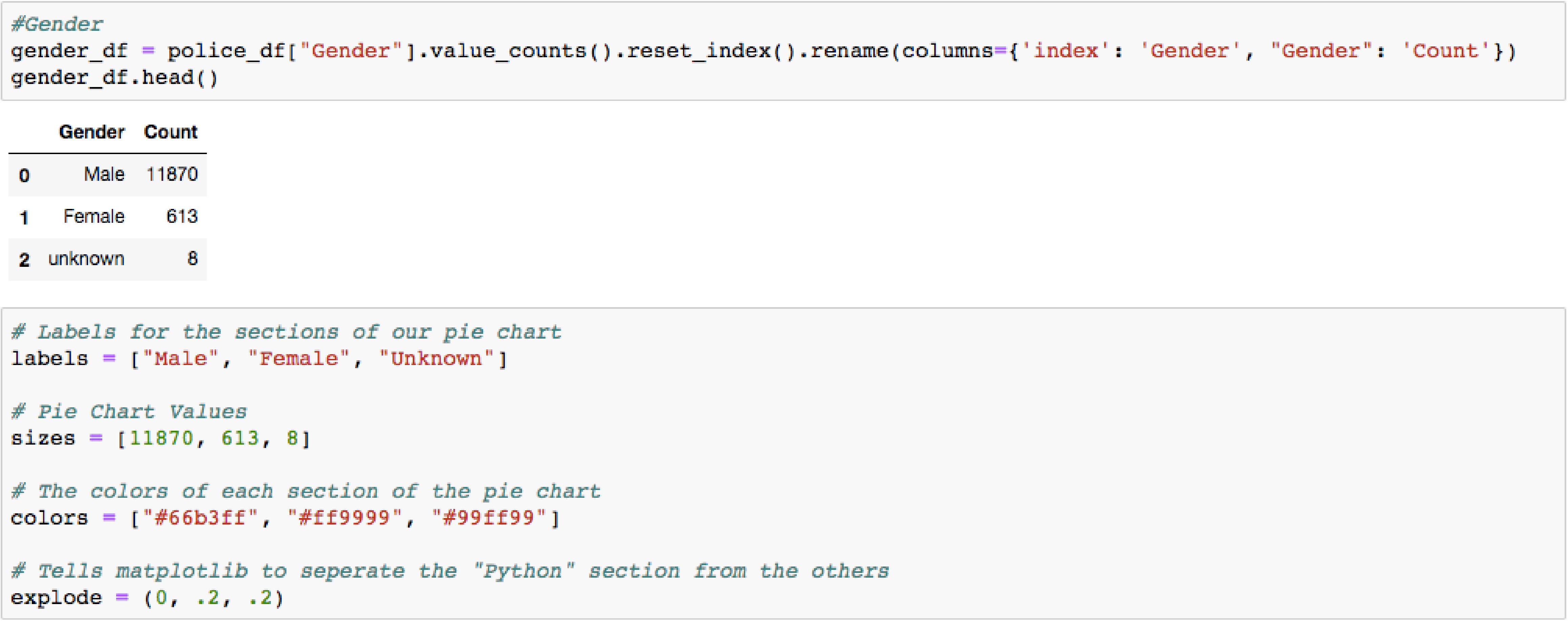


```
#Change Year to dataframe to ascending and remove whitespace from columns
totalpop_df
totalpop_df = totalpop_df.sort_values('Year',ascending=True)
totalpop_df.columns = totalpop_df.columns.str.replace(' ', '')
totalpop_df.head()
```

	Year	Population	Yearly%Change	YearlyChange	
16	2000	282171957	3.48	981144.0	NaN
15	2001	285081556	1.03	2909599.0	NaN
14	2002	287803914	0.95	2722358.0	NaN
13	2003	290326418	0.88	2522504.0	NaN
12	2004	293045739	0.94	2719321.0	NaN

```
#Confirmation that the white space was removed from the columns
print (totalpop_df.columns)
```

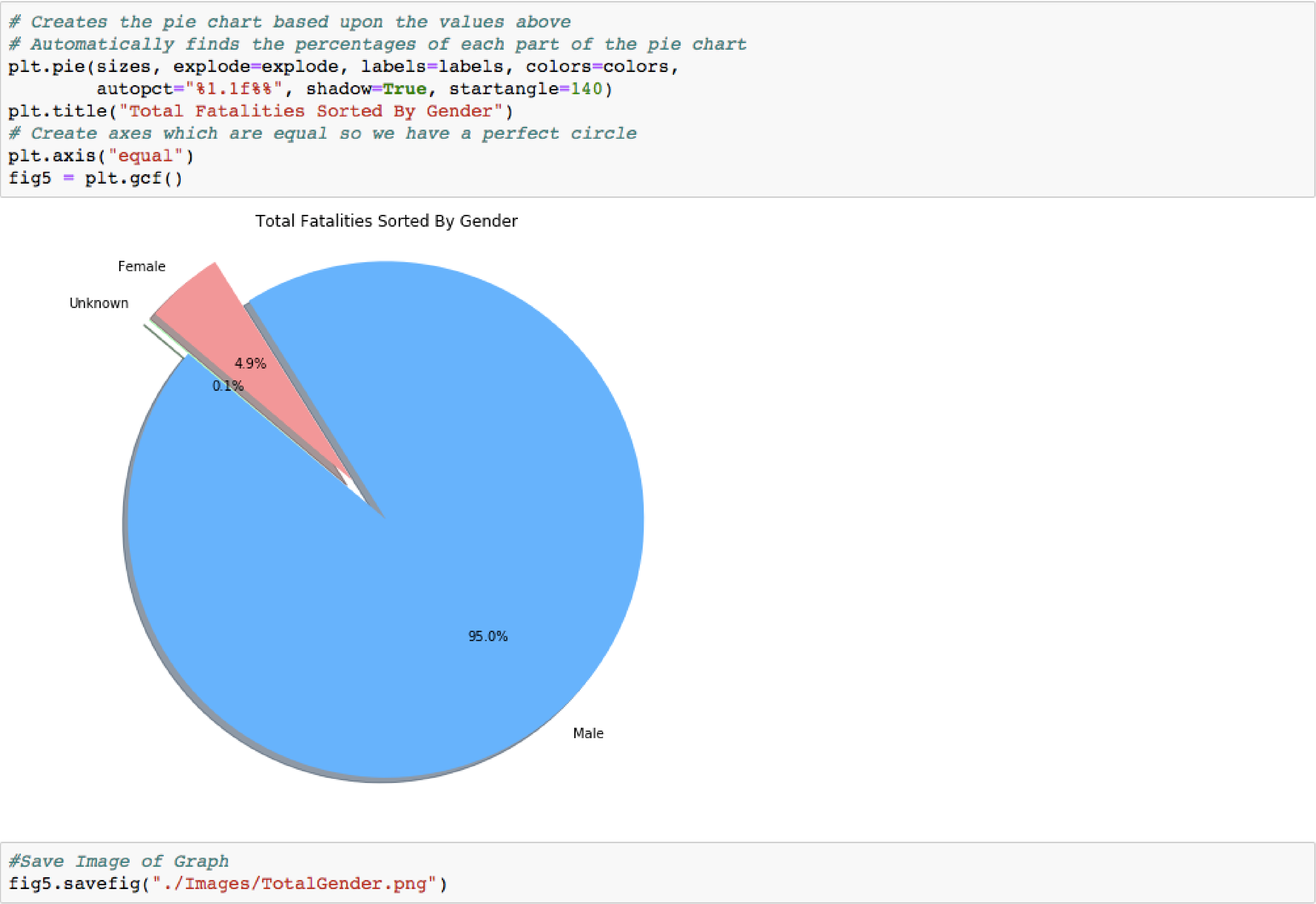
```
Index(['Year', 'Population', 'Yearly%Change', 'YearlyChange', ''], dtype='object')
```



```python
#Gender
gender_df = police_df["Gender"].value_counts().reset_index().rename(columns={'index': 'Gender', "Gender": 'Count'})
gender_df.head()
```

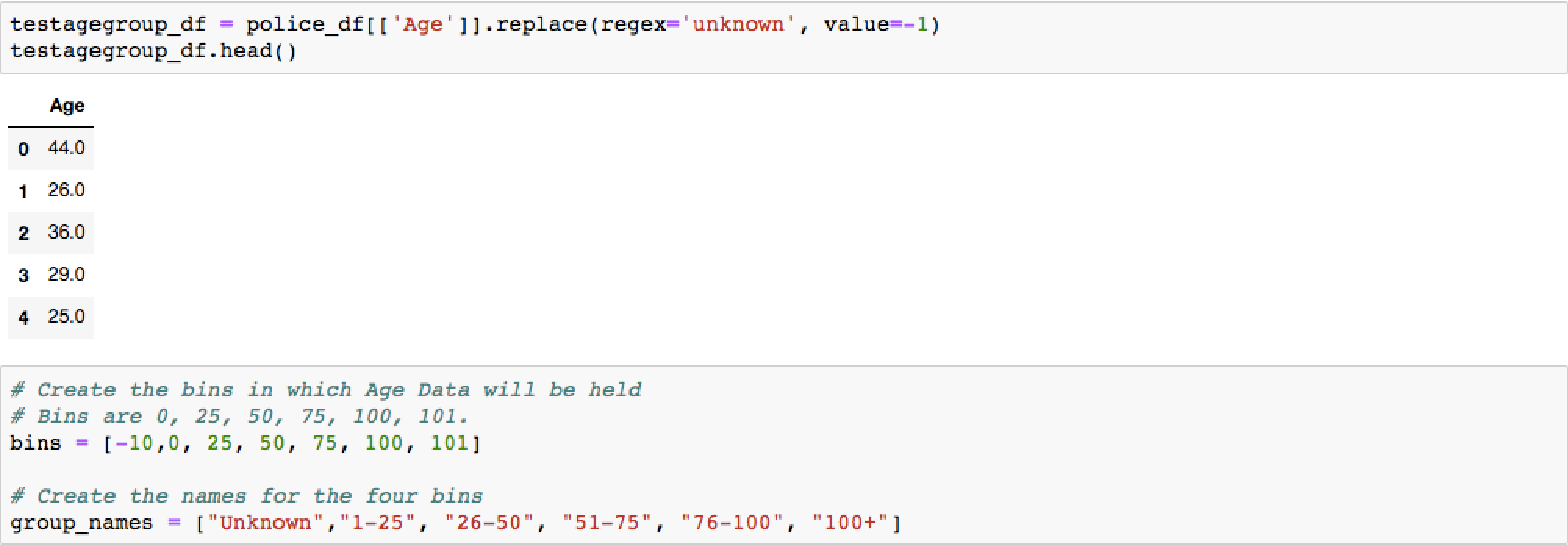
	Gender	Count
0	Male	11870
1	Female	613
2	unknown	8

```python
# Labels for the sections of our pie chart
labels = ["Male", "Female", "Unknown"]

# Pie Chart Values
sizes = [11870, 613, 8]

# The colors of each section of the pie chart
colors = ["#66b3ff", "#ff9999", "#99ff99"]

# Tells matplotlib to seperate the "Python" section from the others
explode = (0, .2, .2)
```



```
# Creates the pie chart based upon the values above
# Automatically finds the percentages of each part of the pie chart
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct="%1.1f%%", shadow=True, startangle=140)
plt.title("Total Fatalities Sorted By Gender")
# Create axes which are equal so we have a perfect circle
plt.axis("equal")
fig5 = plt.gcf()
```

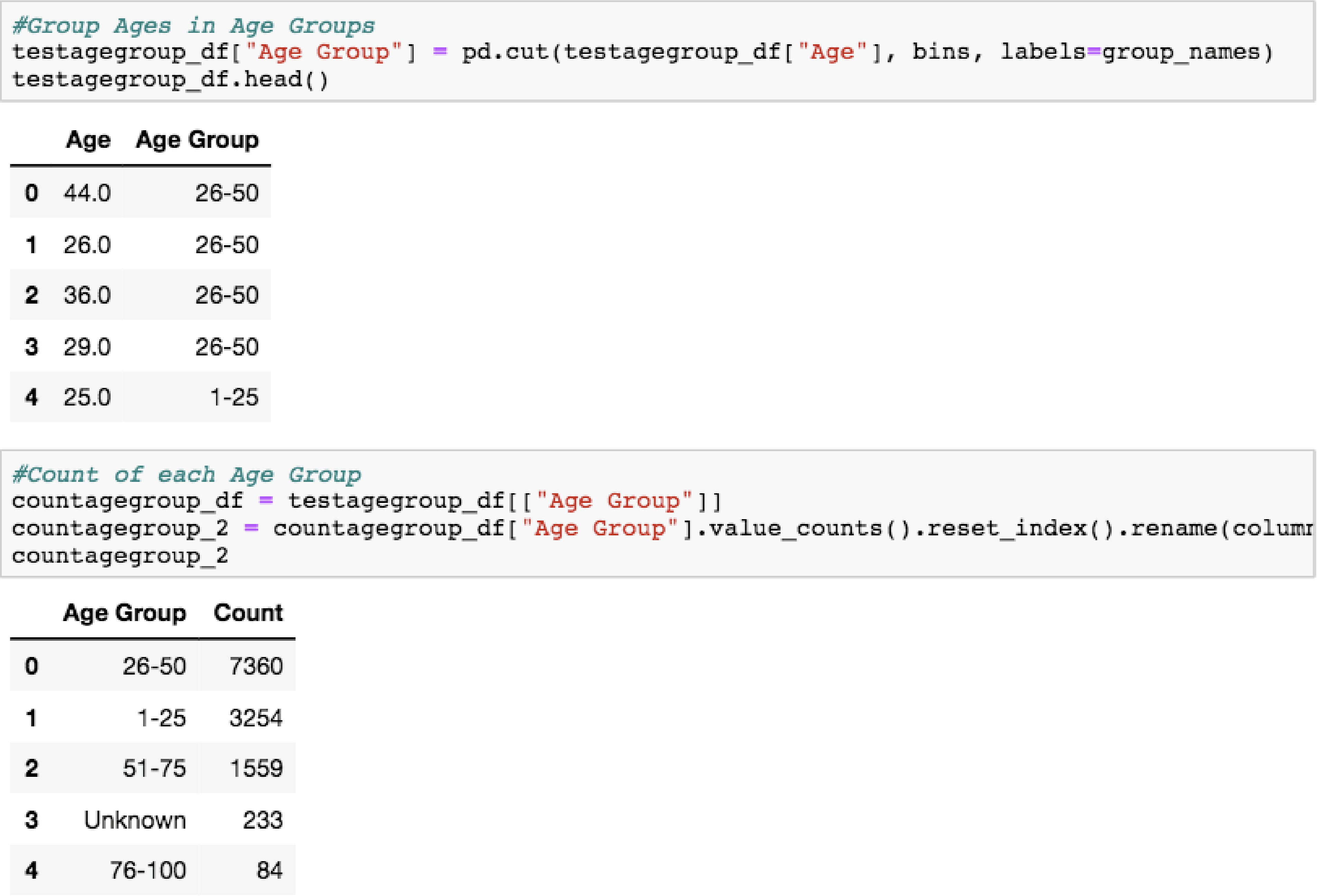
```
#Save Image of Graph
fig5.savefig("./Images/TotalGender.png")
```



```
testagegroup_df = police_df[['Age']].replace(regex='unknown', value=-1)
testagegroup_df.head()
```

	Age
0	44.0
1	26.0
2	36.0
3	29.0
4	25.0

```
# Create the bins in which Age Data will be held
# Bins are 0, 25, 50, 75, 100, 101.
bins = [-10,0, 25, 50, 75, 100, 101]

# Create the names for the four bins
group_names = ["Unknown","1-25", "26-50", "51-75", "76-100", "100+"]
```

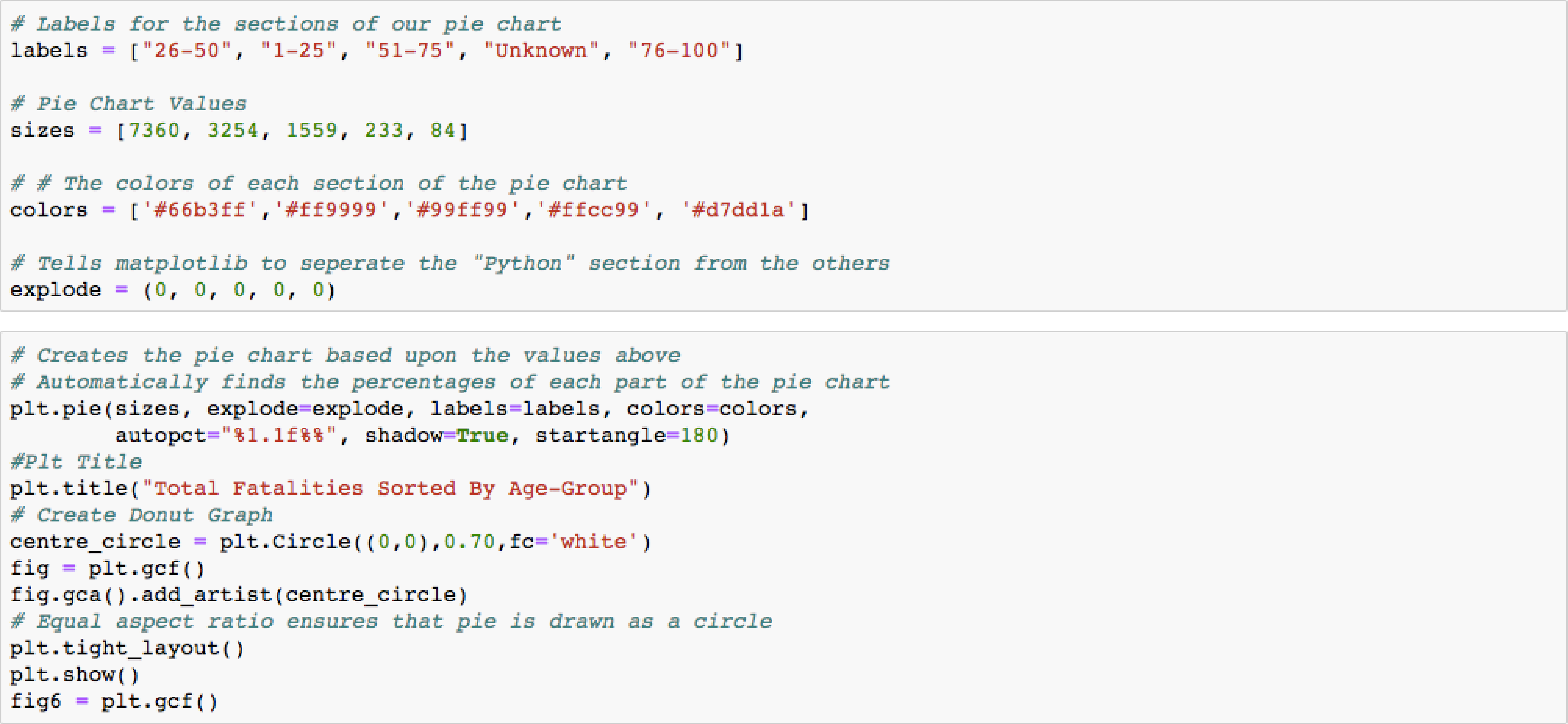


```python
#Group Ages in Age Groups
testagegroup_df["Age Group"] = pd.cut(testagegroup_df["Age"], bins, labels=group_names)
testagegroup_df.head()
```

	Age	Age Group
0	44.0	26-50
1	26.0	26-50
2	36.0	26-50
3	29.0	26-50
4	25.0	1-25

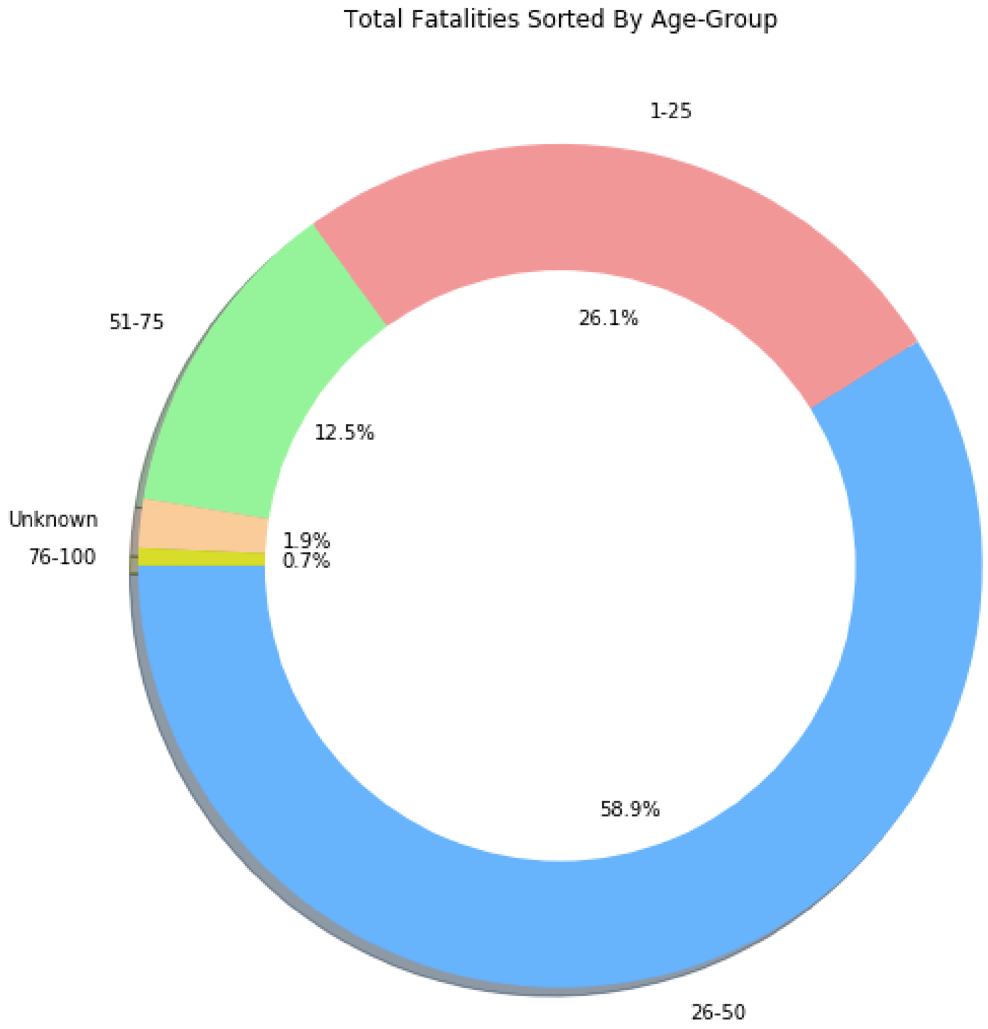
```python
#Count of each Age Group
countagegroup_df = testagegroup_df[["Age Group"]]
countagegroup_2 = countagegroup_df["Age Group"].value_counts().reset_index().rename(colum
countagegroup_2
```

	Age Group	Count
0	26-50	7360
1	1-25	3254
2	51-75	1559
3	Unknown	233
4	76-100	84



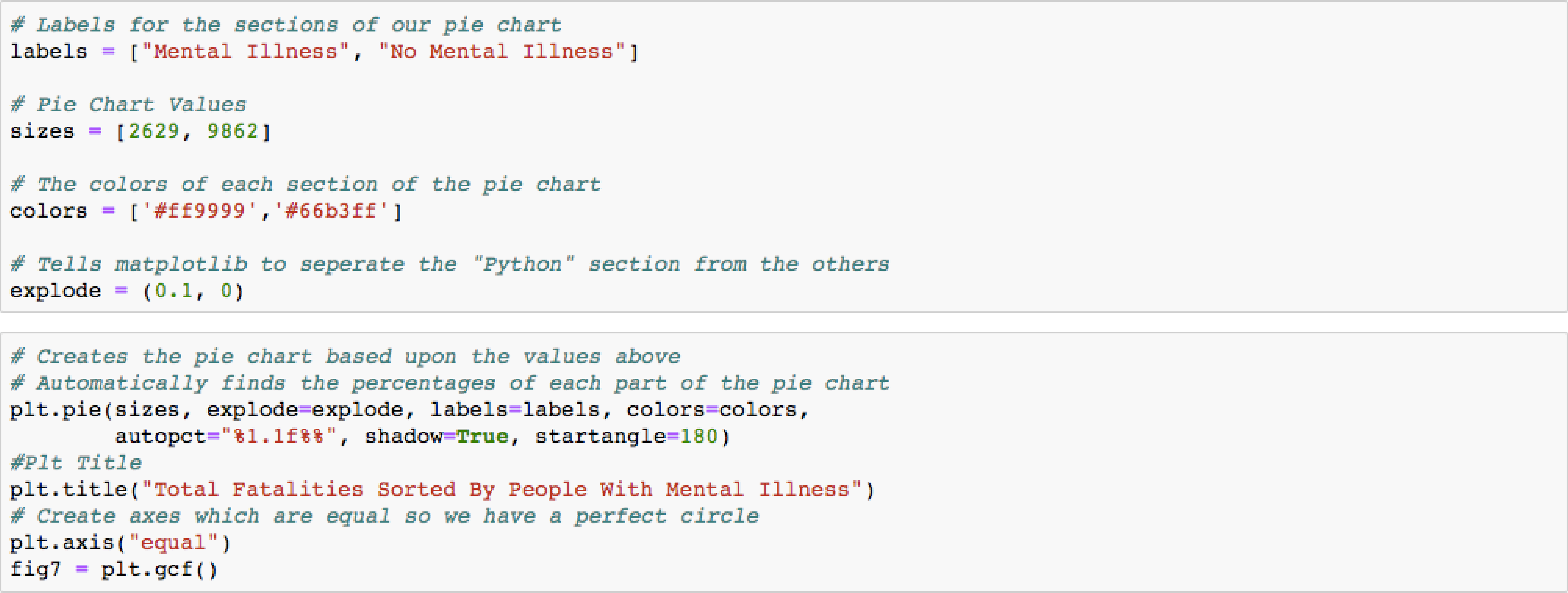
```python
# Labels for the sections of our pie chart
labels = ["26-50", "1-25", "51-75", "Unknown", "76-100"]

# Pie Chart Values
sizes = [7360, 3254, 1559, 233, 84]

# # The colors of each section of the pie chart
colors = ['#66b3ff','#ff9999','#99ff99','#ffcc99', '#d7dd1a']

# Tells matplotlib to seperate the "Python" section from the others
explode = (0, 0, 0, 0, 0)
```

```python
# Creates the pie chart based upon the values above
# Automatically finds the percentages of each part of the pie chart
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct="%1.1f%%", shadow=True, startangle=180)
#Plt Title
plt.title("Total Fatalities Sorted By Age-Group")
# Create Donut Graph
centre_circle = plt.Circle((0,0),0.70,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
# Equal aspect ratio ensures that pie is drawn as a circle
plt.tight_layout()
plt.show()
fig6 = plt.gcf()
```



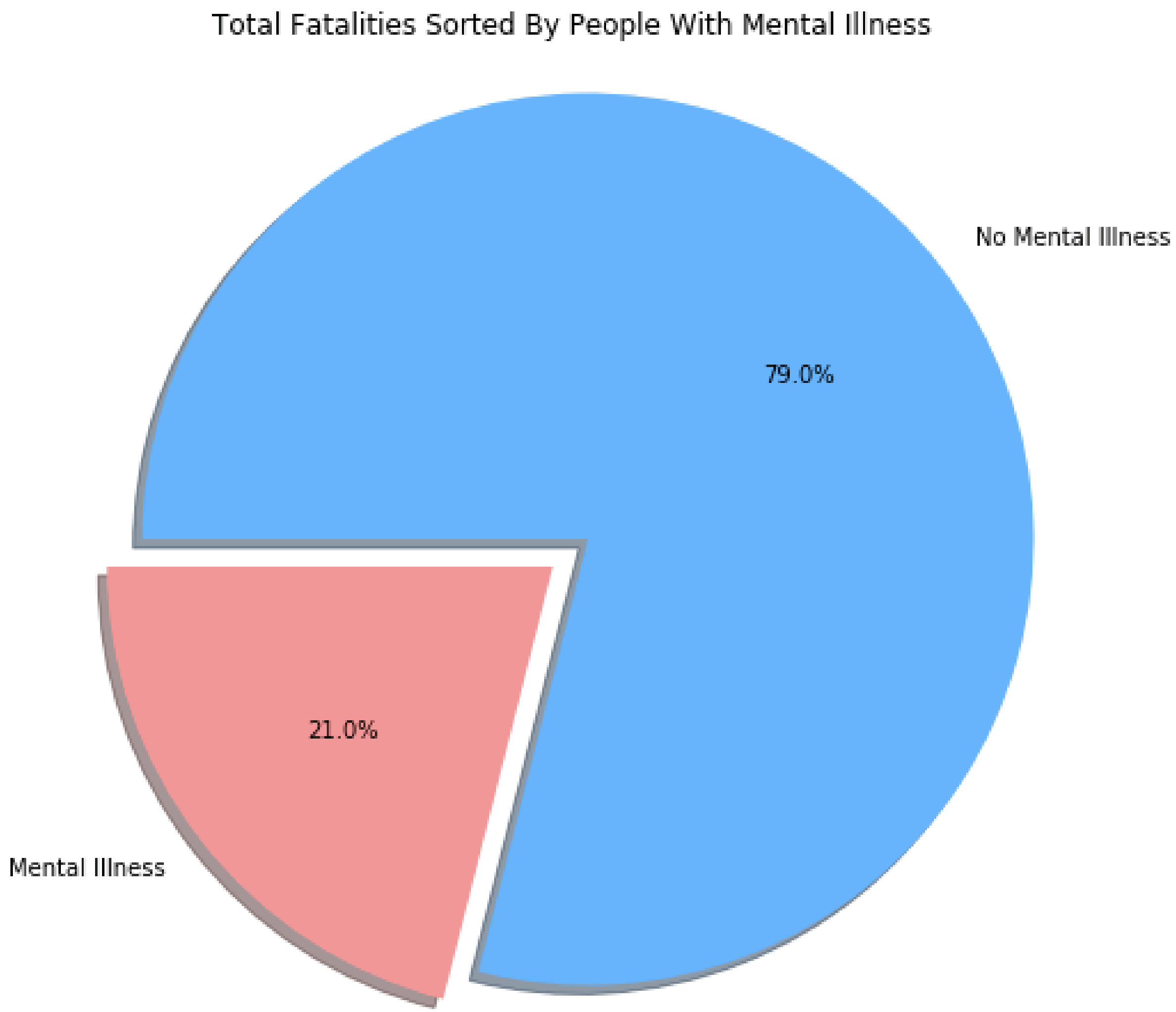
```
<Figure size 720x576 with 0 Axes>
```

```
#Save Image of Graph
fig6.savefig("./Images/TotalAgeGroup.png")
```



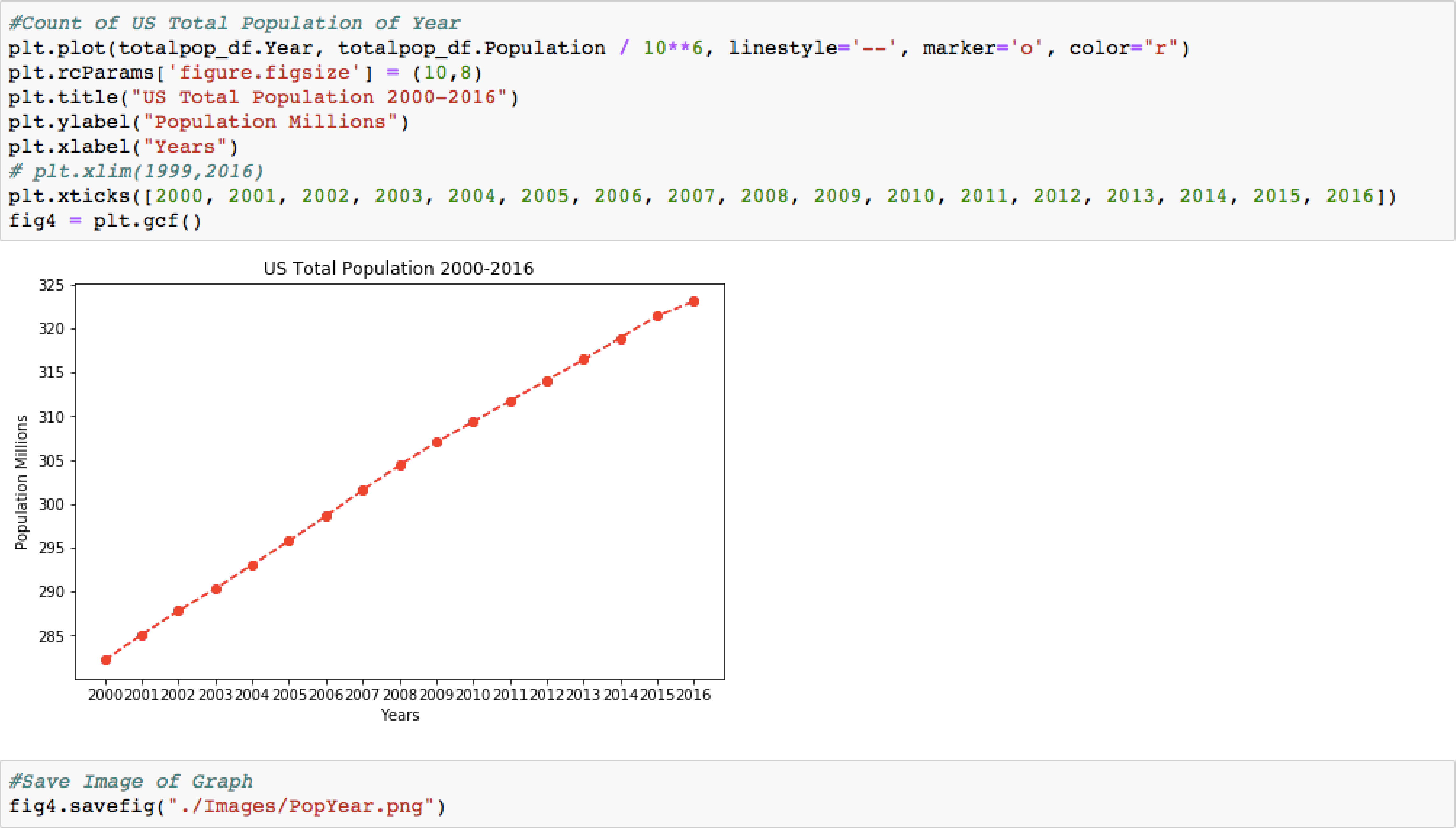
```
# Labels for the sections of our pie chart
labels = ["Mental Illness", "No Mental Illness"]

# Pie Chart Values
sizes = [2629, 9862]

# The colors of each section of the pie chart
colors = ['#ff9999','#66b3ff']

# Tells matplotlib to seperate the "Python" section from the others
explode = (0.1, 0)
```

```
# Creates the pie chart based upon the values above
# Automatically finds the percentages of each part of the pie chart
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct="%1.1f%%", shadow=True, startangle=180)
#Plt Title
plt.title("Total Fatalities Sorted By People With Mental Illness")
# Create axes which are equal so we have a perfect circle
plt.axis("equal")
fig7 = plt.gcf()
```



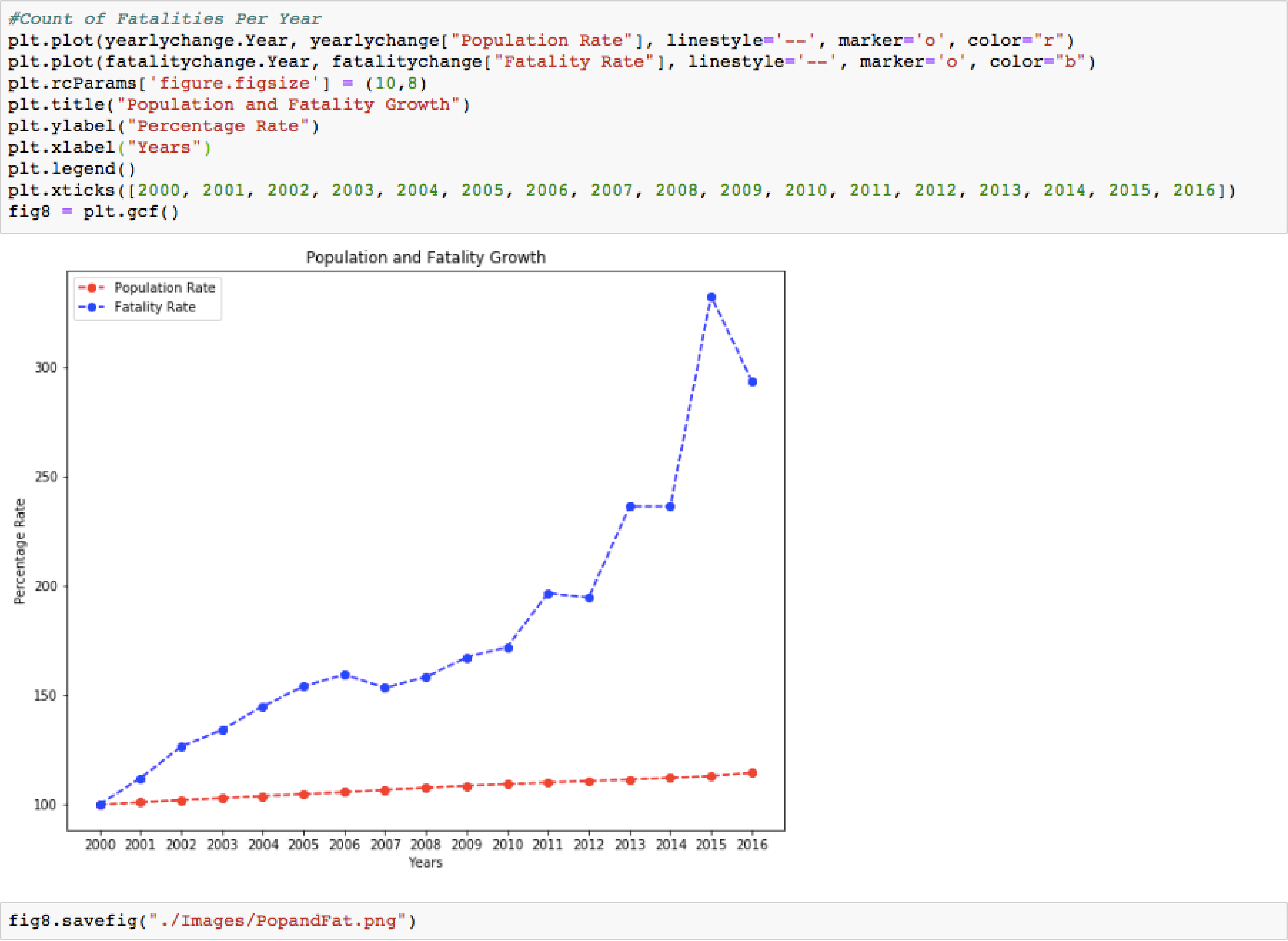
Total Fatalities Sorted By People With Mental Illness

```
#Save Image of Graph
fig7.savefig("./Images/MentalIllness.png")
```



```
#Count of US Total Population of Year
plt.plot(totalpop_df.Year, totalpop_df.Population / 10**6, linestyle='--', marker='o', color="r")
plt.rcParams['figure.figsize'] = (10,8)
plt.title("US Total Population 2000-2016")
plt.ylabel("Population Millions")
plt.xlabel("Years")
# plt.xlim(1999,2016)
plt.xticks([2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016])
fig4 = plt.gcf()
```

```
#Save Image of Graph
fig4.savefig("./Images/PopYear.png")
```



```
#Count of Fatalities Per Year
plt.plot(yearlychange.Year, yearlychange["Population Rate"], linestyle='--', marker='o', color="r")
plt.plot(fatalitychange.Year, fatalitychange["Fatality Rate"], linestyle='--', marker='o', color="b")
plt.rcParams['figure.figsize'] = (10,8)
plt.title("Population and Fatality Growth")
plt.ylabel("Percentage Rate")
plt.xlabel("Years")
plt.legend()
plt.xticks([2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016])
fig8 = plt.gcf()
```

```
fig8.savefig("./Images/PopandFat.png")
```

Questions?