Final_Project

December 9, 2020

[1]:
```
import numpy as np
import pandas as pd

import plotly.graph_objects as go
import plotly.express as px

import matplotlib.pyplot as plt
import seaborn as sns

from datetime import timedelta
from datetime import datetime

pd.options.display.max_columns = None
pd.options.display.max_rows = None
```

[2]:
```
# Import data
# Shooting data
shooting = pd.read_csv('./shootings.csv')
shooting['date'] = pd.to_datetime(shooting.date, format='%Y-%m-%d')

# Race data
#https://www.kff.org/other/state-indicator/distribution-by-raceethnicity/?
 ↪dataView=1&currentTimeframe=0&sortModel=%7B%22colId%22:
 ↪%22Location%22,%22sort%22:%22asc%22%7D
race = pd.read_csv('./race.csv')
race.columns =␣
 ↪['Location','White','Black','Hispanic','Asian','Native','Pacific','Multiple']
race['Other'] = race.Pacific + race.Multiple
race = race.drop(['Pacific','Multiple'], axis=1)
```

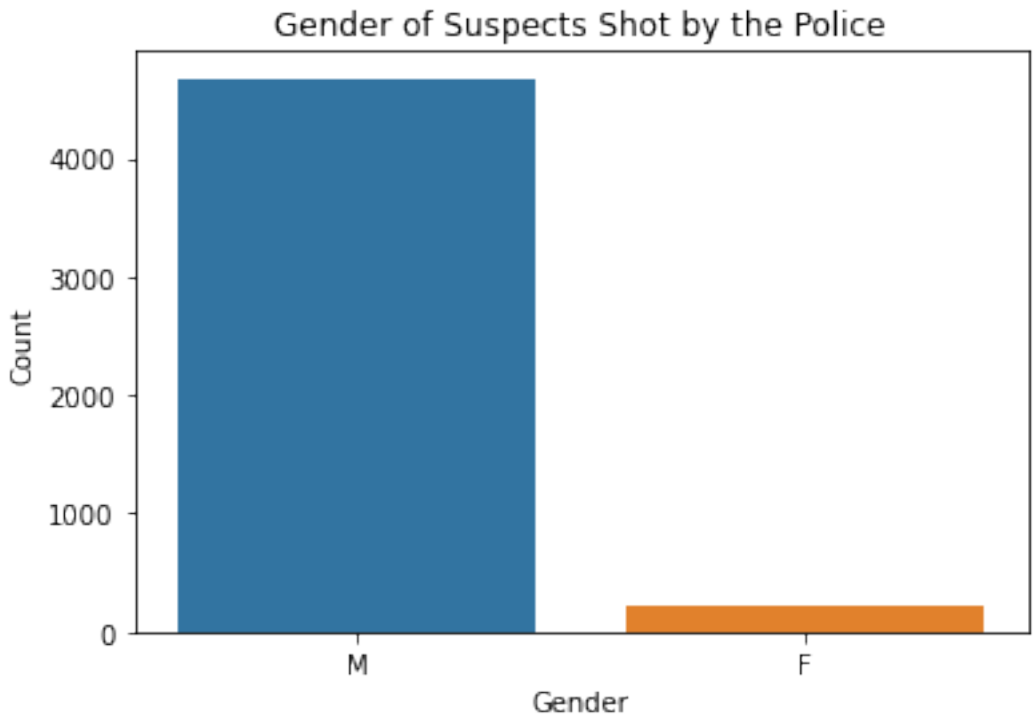
[3]:
```
shooting.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4895 entries, 0 to 4894
Data columns (total 15 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   id                       4895 non-null   int64
```

```
 1   name                       4895 non-null   object
 2   date                       4895 non-null   datetime64[ns]
 3   manner_of_death            4895 non-null   object
 4   armed                      4895 non-null   object
 5   age                        4895 non-null   float64
 6   gender                     4895 non-null   object
 7   race                       4895 non-null   object
 8   city                       4895 non-null   object
 9   state                      4895 non-null   object
 10  signs_of_mental_illness    4895 non-null   bool
 11  threat_level               4895 non-null   object
 12  flee                       4895 non-null   object
 13  body_camera                4895 non-null   bool
 14  arms_category              4895 non-null   object
dtypes: bool(2), datetime64[ns](1), float64(1), int64(1), object(10)
memory usage: 506.8+ KB
```

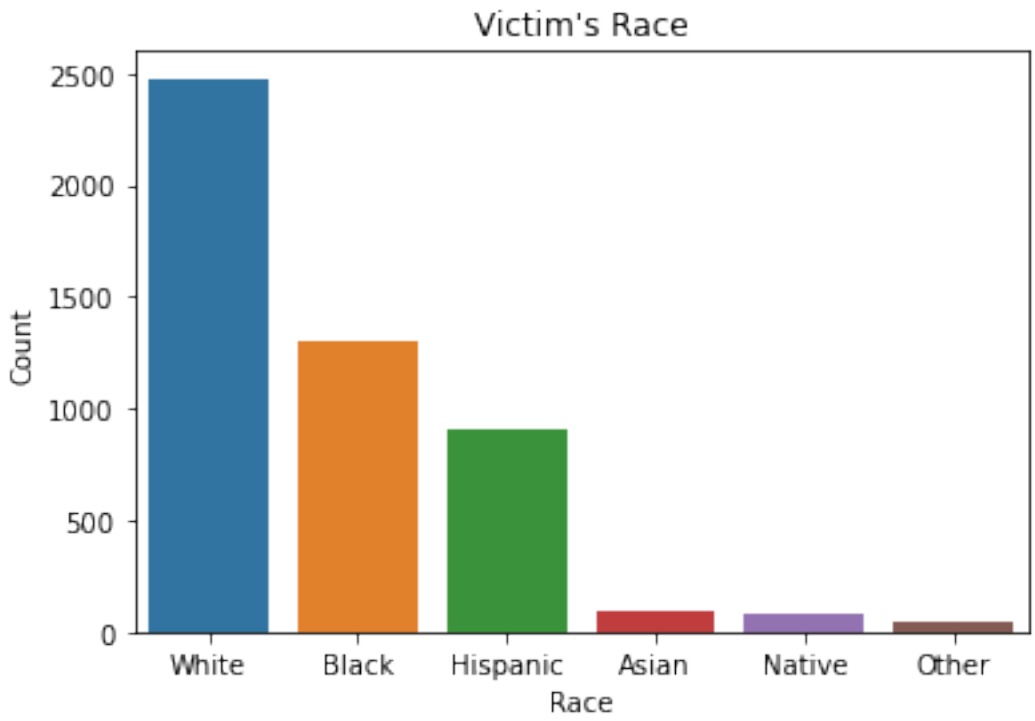
[4]:
```
sns.countplot('gender', data = shooting)
plt.title('Gender of Suspects Shot by the Police')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.show()
```

```
/Applications/anaconda3/lib/python3.7/site-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or
misinterpretation.
  FutureWarning
```



[5]:
```
race_distri = shooting['race'].value_counts().reset_index()
race_distri.columns = ['race', 'Count']
race_distri.sort_values('Count', ascending = False, inplace = True)

sns.barplot(x = 'race', y = 'Count', data = race_distri)
plt.title("Victim's Race")
plt.xlabel("Race")
plt.ylabel('Count')
plt.show()
```



[6]:
```
race_distri = shooting['race'].value_counts().reset_index()
race_distri.columns = ['race', 'Shooting']
race_count = race_distri.Shooting.sum()

race_us = race.drop('Location',axis=1).sum(axis=0)
total_pop = race_us.sum()

race_us = pd.DataFrame(race_us).reset_index()
race_us.columns = ['race', 'Total Population']

shooting_by_race = pd.merge(race_us,race_distri,how='inner',on='race')

shooting_by_race['Total Population'] = shooting_by_race['Total Population'] /␣
↪total_pop
shooting_by_race['Shooting'] = shooting_by_race['Shooting'] / race_count

shooting_by_race = shooting_by_race.transpose()
shooting_by_race.columns = shooting_by_race.iloc[0]


shooting_by_race = shooting_by_race.drop('race',axis=0).reset_index()
shooting_by_race.columns =␣
↪['Quantity','White','Black','Hispanic','Asian','Native','Other']
```

```
shooting_by_race = pd.melt(shooting_by_race,
                           id_vars=['Quantity'],
                           ␣
 ↪value_vars=['White','Black','Hispanic','Asian','Native','Other'],
                           value_name='Percentage')

shooting_by_race.columns = ['Quantity', 'Race', 'Percentage']
fig = px.bar(shooting_by_race, x="Quantity", y="Percentage", color="Race",␣
 ↪title="US Population Demographic vs Victim's Demographic")
fig.show()
```

[7]:
```
shootings = shooting[shooting['age'] != 37.11793090137039]

fig_age = px.histogram(shootings, x="age", color="age", width = 700, height =␣
 ↪350)
fig_age.update_layout(title = {'text':"Age of victims",'y':0.95,
            'x':0.5, 'xanchor':'center', 'yanchor':'top'},  xaxis_title="Age",
        yaxis_title="Count",
                    font=dict(
            family="Courier New, monospace",
            size=15,
            color="RebeccaPurple"
        ))

fig_age.show()
```

[8]:
```
shooting_2 = shooting[shooting['age'] != 37.11793090137039]
state_age_count = shooting_2[['age', 'race']]
state_age_count.head()

state_age_count_1 = pd.DataFrame(state_age_count.groupby('age')['race'].
 ↪value_counts())
state_age_count_1

state_age_count_1.rename(columns = {'race' : 'Count'}, inplace = True)

state_age_count_2 = state_age_count_1.reset_index()
state_age_count_2.rename(columns = {'race' : 'Race'}, inplace = True)
state_age_count_2.sort_values('Race', ascending= True, inplace = True)

fig_state = px.bar(state_age_count_2, y="Count", x="age", color='Race',␣
 ↪orientation='v',
             hover_data=["Race", "Count"], height=600, width = 1000)

fig_state.update_layout(title = {'text': "Race distribution at different␣
 ↪Age",'y':0.99,
```

```
            'x':0.54, 'xanchor':'center', 'yanchor':'top'},  xaxis_title="Age",
        yaxis_title="Count",
                    font=dict(
            family="Courier New, monospace",
            size=16,
            color="RebeccaPurple"
        ))
fig_state.show()
```

[9]:
```
state_race_count = shooting[['state', 'race']]
state_race_count.head()

state_race_count_1 = pd.DataFrame(state_race_count.groupby('state')['race'].
 ↪value_counts())
state_race_count_1

state_race_count_1.rename(columns = {'race' : 'Count'}, inplace = True)

state_race_count_2 = state_race_count_1.reset_index()
state_race_count_2.rename(columns = {'race' : 'Race'}, inplace = True)
state_race_count_2.sort_values('state', ascending= False, inplace = True)
state_race_count_3 = state_race_count_2
order = np.array(state_race_count_3.groupby('state').sum().reset_index().
 ↪sort_values('Count',ascending=True).state)
fig_state = px.bar(state_race_count_2, x="Count", y="state", color='Race',␣
 ↪orientation='h',
             hover_data=["Race", "Count"], height=1200, width = 800)

fig_state.update_layout(title = {'text': "State wise distribution of victim's␣
 ↪race",'y':0.99,
            'x':0.54, 'xanchor':'center', 'yanchor':'top'}, ␣
 ↪xaxis_title="Count",
                      yaxis= {
        'categoryorder': 'array',
        'categoryarray': order
    },
        yaxis_title="state",
                    font=dict(
            family="Courier New, monospace",
            size=16,
            color="RebeccaPurple"
        ))
fig_state.show()
```

[10]:
```
state_race_count_3 = state_race_count_2
```

```
order = np.array(state_race_count_3.groupby('state').sum().reset_index().
↪sort_values('Count',ascending=True).state)
```

[11]:
```
type_of_arm = shooting['armed'].value_counts().reset_index()
type_of_arm.columns = ['Armed with', 'Count']
type_of_arm = type_of_arm[type_of_arm['Armed with'] != 'unknown']

fig_arm = px.bar(type_of_arm.head(5), x = 'Count', y = 'Armed with', color =␣
↪'Armed with',
                 width = 700, height = 350,hover_name= 'Armed with',␣
↪orientation= 'h')

fig_arm.update_layout(title = {'text':"Top 5 weapons",'y':0.95,
            'x':0.5, 'xanchor':'center', 'yanchor':'top'},  xaxis_title="Count",
        yaxis_title="Armed with",
                     font=dict(
            family="Courier New, monospace",
            size=15,
            color="RebeccaPurple"
        ))
fig_arm.show()
```