FinalProject_group062

March 20, 2020

1 COGS 108 - Final Project

1.1 Permissions

Place an X in the appropriate bracket below to specify if you would like your group's project to be made available to the public. (Note that PIDs will be scraped from the public submission, but student names will be included.)

[x] YES - make available [ ] NO - keep private

1.2 Overview

The goal of this project was to measure and analyze the distribution of San Diego police traffic stops along the lines of race of the subject, time of day of the stop, and region the stop takes place in. More importantly, we wanted to find which of these factors points most clearly to a correlation between its own distribution and the police stop. In order to do this, we took public data from 2010 to 2017 about San Diego police traffic stops, compiled it all together, then divided it along the lines of subject race, region, and time of day, separately, then compared it to population demographics in San Diego in order to draw a conclusion. Analyzing this information allowed us to find the impact of each of these factors, and how they contribute to the frequency of police stops. In the end, we found the strongest relationship between a person's gender and their likeliness to be pulled over.

1.3 Names

- Nevan Samadhana
- Matthew Crooks
- Kelvin Nguyen
- Shannon He
- Nicholas Klopp

1.4 Group Members IDs

- A13837722
- A13760423
- A13431011
- A13377202

- A15753860

1.5 Research Question

We are interested in seeing who the San Diego police department pull over more. We wanted to find if a person's trait such as gender, race, or age affected their pullover rate or if other factors did such as time of day or day of month. We wanted to find ***which single variable contributes most to the probability that an individual is pulled over by San Diego police for a traffic stop?***

1.6 Background and Prior Work

Ever since high school, our group has seen news articles pop up and attract widespread attention regarding police traffic stops. Therefore, we wondered if we could dedicate this project towards finding an answer to the question: is the motivation behind pulling citizens over purely innocuous within San Diego county? According to Voice of San Diego and the UC San Diego Extension Center for Research black people were being discriminated against. They found that black people were stopped at a disproportionate rate. They found that even though Blacks only made up four percent of San Diego's population they made up eight percent of the people being stopped overall by police. This was in the time period between July 2018 and July 2019 that included 41,036 stops made by police. This is compared to whites and hispanics who were stopped proportionately to their population. Hispanics make up 31 percent of San Diego's population but only made up 29 percent of the police stops. Whites make up 54 percent of San Diego's population and made up 54 percent of the police stops. They also found that blacks were being searched at a higher rate compared to other ethnic backgrounds. The Voice of San Diego and UC San Diego Extension Center found that of the police stops they were being searched twenty-two percent of the time. This is compared to whites who were being searched only seventeen percent of the time. They also found that when they searched black people property seizures were lower at only fourteen percent compared to white that had their property seized eighteen percent of the time.(1)

As a result they concluded that there is bias being made against black people in San Diego county. Another report done by the ACLU also found that there were discriminatory patterns being incited by the San Diego police department. They found that "San Diego police are 219% more likely to stop black people than white people and 25% more likely to search and 59% more likely to use force against black people during these stops". The San Diego police says this is because Blacks make up more of the homeless population. ACLU also found that "black people were 163% more likely to be stopped for a traffic violation than white people and 86% more likely to be stopped for moving violations specifically (i.e., actions like speeding or running a red light). This suggests black people were more likely than white people to be stopped on the street and also more likely than white people to be stopped in traffic". These findings align with our group's belief that blacks are being discriminated against by the police. We will investigate whether we find the same findings as these reports and on top of that we will investigate What factors contribute towards the highest probability of being pulled over for a traffic stop? Does the subject race/time of day/region affect pullover rate and which factor contributes most towards pullover rate?(2)

References (include links):

1) https://www.voiceofsandiego.org/topics/public-safety/san-diego-law-enforcement-searches-blacks-more-finds-contraband-on-them-less/
2) https://www.sandiegouniontribune.com/opinion/commentary/story/2019-12-18/san-diego-police-bias-aclu-report

1.7 Hypothesis

Out of all of the factors, we believe that the strongest relationship exists between subject race and traffic stops.

We proposed this hypothesis because, we are constantly inundated with stories about black men being targeted by police like those in Ferguson, Missouri and Eric Garner in New York. Regardless of what side of the political spectrum you sit on, it is impossible to escape these news stories, and therefore, we have been conditioned to think that a relationship does exist between subject race and their likelihood of being pulled over.

1.8 Dataset(s)

Dataset Name: Crime Data from 2010 to Present Link to the dataset: https://data.lacity.org/api/views/63jg-8b9z/rows.csv?accessType=DOWNLOAD Number of observations: 2.09 million Description: LA city's dataset of crime starting in 2010. It includes all of the crime reports made within LA city during that time.

Dataset Name: Traffic stop data 2014 Link to the dataset: https://data.sandiego.gov/datasets/police-vehicle-stops/ Number of observations: 144164 Description: SD city's dataset of traffic stops in 2014. It includes all of the police reports of traffic stops. The dataset has variables such as stop_id, stop_cause, service_area, subject_race, subject_sex, subject_age, date_time, date_stop, time_stop, sd_resident, arrested, searched, obtained_consent, contraband_found, and property_seized.

Dataset Name: Traffic stop data 2015 Link to the dataset: https://data.sandiego.gov/datasets/police-vehicle-stops/ Number of observations: 115422 Description: SD city's dataset of traffic stops in 2015. It includes all of the police reports of traffic stops. The dataset has variables such as stop_id, stop_cause, service_area, subject_race, subject_sex, subject_age, date_time, date_stop, time_stop, sd_resident, arrested, searched, obtained_consent, contraband_found, and property_seized.

Dataset Name: Traffic stop data 2016 Link to the dataset: https://data.sandiego.gov/datasets/police-vehicle-stops/ Number of observations: 103051 Description: SD city's dataset of traffic stops in 2016. It includes all of the police reports of traffic stops. The dataset has variables such as stop_id, stop_cause, service_area, subject_race, subject_sex, subject_age, date_time, date_stop, time_stop, sd_resident, arrested, searched, obtained_consent, contraband_found, and property_seized.

Dataset Name: Traffic stop data 2017 Link to the dataset: https://data.sandiego.gov/datasets/police-vehicle-stops/ Number of observations: 103362 Description: SD city's dataset of traffic stops in 2017. It includes all of the police reports of traffic stops. The dataset has variables such as stop_id, stop_cause, service_area, subject_race,

subject_sex, subject_age, date_time, date_stop, time_stop, sd_resident, arrested, searched, obtained_consent, contraband_found, and property_seized.

Dataset Name: U.S. Census Bureau San Diego Demographic Table Link to the dataset: https://www.census.gov/quickfacts/fact/table/sandiegocountycalifornia,CA/PST045218 Data used: Gender breakdown and Race breakdown In this case, it was difficult to translate this website into a csv or easily digestible data file such as csv or json. Therefore, because we took a very minimal amount of data off of the website (<10 values), we found that it was much easier to just hard code these values into our project. While not correct in every sense, this was the best decision in order to maximize our time spent on data analysis.

Because we are using multiple datasets, we stored each csv file as a dataframe, and then concatenated all of dataframes into one.The dataset we are using is a public data set created by the San Diego Police Department so they can analyze if any discrimination is occuring within San Diego. Specific variables that we would like to examine would include the police stop id, stop cause, service area, subject race, subject sex, subject age, date of the police stop, whether the person was a San Diego resident or not, whether they were arrested or not, whether they were searched or not, the division where they were stoped and their main race (we grouped the races into four major groups White, Asian, Hispanic, and Blacks). The sample size that spans several years from 2014-2017. This data has been collected through sources that we have found on San Diego's public data site and have been generally "clean" CSV databases.

1.9 Setup

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
from scipy.stats import chisquare
from scipy import stats
from scipy.stats import ttest_ind, chisquare, normaltest
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import statsmodels.api as sm
```

1.10 Data Cleaning

- The datasets that we are working with are not very clean and require quite a bit of standardizing.
- San Diego police officers make frequent mistakes due to human error when inputting details for their traffic stops.
- A lot of the "uncleanliness" lies behind the unique ways of identifying the same information, ie. 'female', 'f', 'F' being entered by police when recording the gender. For that reason, you can see below that our primary focus when we started cleaning was to standardize values given identical information but different ways of writing it.

- A few of the columns, because they were in large part left blank by police when originally filling out a report (ie. obtained consent), we decided to drop because the data we procured wouldn't be representative of the entire sample. Also once we ran our data cleaning methods to standardize the data, we didn't run into any further problems with any null values.
- The date of the traffic stop is stripped, and two new columns are created to analyze the day and month of those stops.
- A new "division" column is also created which maps the integer division of the stop to its respective geographic location i.e "110" means the stop was performed in the "Northern" division of San Diego county.
- We standardized the subject_race in order to focus on a few "main" races. Asian races such as "Laotian" or "Japanese" were mapped into "Asian", and we mapped low frequency minority groups who were not Black, Hispanic, or Asian into "Other".

[2]:
```
# Load datasets from 2014-2017
year_2014 = pd.read_csv('vehicle_stops_2014_datasd_v1.csv')
year_2015 = pd.read_csv('vehicle_stops_2015_datasd_v1.csv')
year_2016 = pd.read_csv('vehicle_stops_2016_datasd_v1.csv')
year_2017 = pd.read_csv('vehicle_stops_2017_datasd_v1.csv')

# Combine datasets into a list
pd_list = [year_2014,year_2015,year_2016,year_2017]
all_years = pd.concat(pd_list)
```

We chose to standardize the data for the age column of our data to remove extreme outliers that were most likely mistakes on data entry. For example, we had some values that were as high as 1000 which makes little sense.

[3]:
```
# Standardized function for age
def clean_age(string):
    string = str(string)
    if string == 'nan':
        return np.nan
    pattern = re.compile(r'[^\d+]')
    string = pattern.sub('',string)
    if string.isnumeric() == False:
        return np.nan
    string = string.strip()
    string = int(string)
    if string >= 16 and string < 100:
        return string
    else:
        return np.nan
```

[4]:
```
# Standardizr 'age' column
all_years['subject_age'] = all_years['subject_age'].apply(clean_age)
```

Standardize the reasons for being pulled over in a traffic stop, making sure that there aren't over 10 different ways of not specifying a reason.

[5]:
```
#Standardize function for stop cause
def clean_stop_cause(string):
    string = str(string)
    notMarked = ["not secified", "NOT SPECIFIED",'NOT CHECKED','No Cause␣
 ↪Specified on a Card','not marked not marked', 'NOTHING MARKED', 'not␣
 ↪marked','none listed','not noted','not listed','no cause listed', 'not␣
 ↪marked  not marked','none noted', 'CAUSE NOT LISTED ACTION NOT LISTED', 'NOT␣
 ↪MARKED', 'nan']
    bicycle = ["Bicycle", "Bicycle Bicycle"]
    pedestrian = ["PedestrianPedestrian"]
    movingViolation = ["Moving Violation", "&Moving Violation"]
    equipmentViolation = ["Equipment Violation", "&Equipment Violation"]
    h_and_s = ['MUNI, County, H&S Code','Muni, County, H&S Code']
    if string in notMarked:
        output = "Not Marked"
    elif string in bicycle:
        output = "Bicycle"
    elif string in pedestrian:
        output = "Pedestrian"
    elif string in movingViolation:
        output = "Moving Violation"
    elif string in equipmentViolation:
        output = "Equipment Violation"
    elif string in h_and_s:
        output = "MUNI, County, H&S Code"
    else:
        output = string

    return output
```

[6]:
```
all_years["stop_cause"] = all_years["stop_cause"].apply(clean_stop_cause)
```

Standardize the subject's race to ensure erroneous values such as numbers and 'bicycle' are taken care of.

[7]:
```
# Standardize 'subject_race' column
def clean_subject_race(string):
    string = str(string)
    none = ["2", "5", "4", "1", "NOT MARKED", "37", "3", "26", "32", "55",␣
 ↪"48", "28", "BICYCLE", "Pedestrian"]
    if string in none:
        return "None"
    else:
        return string
```

[8]:
```
all_years["subject_race"] = all_years["subject_race"].apply(clean_subject_race)
```

[9]:
```
#Standardize date_stop column
def clean_date_stop(date):
    date = date.split('-')
    day = date[-1]
    day = day.strip()
    return day
```

Create a new column in our dataframe for the day of the month that the stop occurred. The values inside this column should only include values from 1-31 which you can see in the next line is exactly what our code achieves.

[10]:
```
all_years['day'] = all_years['date_stop'].apply(clean_date_stop)
```

[11]:
```
def clean_month_stop(date):
    date = str(date)
    date = date.split("-")
    month = date[-2]
    month = month.strip()
    return month
```

Create a new column in our dataframe for the month of the year that the stop occurred. The values inside this column only include values from 1-12(Jan-Dec).

[12]:
```
all_years["month"] = all_years["date_stop"].apply(clean_month_stop)
```

Standardize the SD resident column to only include 'Y', 'N', or no response

[13]:
```
# Standardize 'sd_resident' column
all_years[all_years['sd_resident']== 'y'] = 'Y'
all_years[all_years['sd_resident']== 'n'] = 'N'
all_years[all_years['sd_resident']== ' '] = None
```

Standardize the arrested column to only include 'Y', 'N', or no response

[14]:
```
# Standardize 'arrested' column
all_years[all_years['arrested']== 'y'] = 'Y'
all_years[all_years['arrested']== 'n'] = 'N'
all_years[all_years['arrested']== ' '] = None
all_years[all_years['arrested']== 'b'] = None
all_years[all_years['arrested']== 'M'] = None
```

Standardize the searched column to only include 'Y', 'N', or no response

[15]:
```
# Standardize 'searched' column
all_years[all_years['searched']== 'y'] = 'Y'
all_years[all_years['searched']== 'n'] = 'N'
all_years[all_years['searched']== ' '] = None
```

Since the only data we want are the factors that contribute to a probability of being pulled over at

traffic stops, columns below might not be one of these factors so that we should drop them at this point.

[16]:
```
# Drop all the useless features
all_years = all_years.
 ↪drop(columns=['obtained_consent','contraband_found','property_seized','date_time'],axis=1)
```

Standardize 'division' by changing the area code into general division in San Diego, and map the area code into a new column 'division'. Plus, since all the unknown area codes are not helpful for our analysis, we mark all of these unknown areas with a '0'.

[17]:
```
# Change the area code into general divisions
division = {'110':'Northern','320':'Eastern','610':'Western',"930":
 ↪'NorthWestern','820':'MidCity','710':'Southern','120':'Northern','230':
 ↪'NorthEastern','240':'NorthEastern','720':'Southern',"430":
 ↪'SouthEastern','310':'Eastern','510':'Central',"Unknown":None,"810":
 ↪'MidCity','440':'SouthEastern','830':'MidCity',"520":'Central',"620":
 ↪'Western',"630":"Western","130":'Northern',"530":'Central',"840":'MidCity'}
# Change any unknown area into '0'
all_years['service_area'].loc[all_years.service_area == 'Unknown'] = '0'
# Map the area code into new column 'division'
all_years['division'] = all_years['service_area'].map(division)
```

Standardize 'race' by changing the letter representations into full-named races which makes them easier to identify, and map the original subject_race into a new column 'race'.

[18]:
```
# Gives out the full name of races instead of single letters
race = {"W":"White", "A":"Other Asian","B":"Black","C":"Chinese","D":
 ↪"Cambodian","F":"Filipino","G":"Guamanian","H":"Hispanic","I":"Indian","J":
 ↪"Japanese","K":"Korean","L":"Laotian","O":"Other","P":"Pacific Islander","S":
 ↪"Samoan","U":"Hawaiian","V":"Vietnamese","Z":"Asian Indian","X":"TBD",}
# Map the 'subject_race'(single letter) into new column 'race'
all_years["race"] = all_years["subject_race"].map(race)
```

Transform all races into a few general race groups, and map all the races to a new column 'main_race'; also, drop any missing values in 'main_race' column.

[19]:
```
# Take out the races we want to analyze into general race groups
main_races = {"White":"White", "Other Asian": 'Asian',"Black":"Black","Chinese":
 ↪'Asian',"Cambodian":"Asian","Filipino":'Asian',"Guamanian":
 ↪'Other',"Hispanic":"Hispanic","Indian":'Other',"Japanaese":'Asian',"Korean":
 ↪"Asian","Laotian":'Asian',"Other":'Other',"Pacific Islander":
 ↪'Asian',"Samoan":"Other","Hawaiian":'Other',"Vietnamese":'Asian',"Asian␣
 ↪Indian":'Asian',"TBD":'Other'}
# Map races into a new column for 'main_race' that we want to analyze
all_years["main_race"] = all_years["race"].map(main_races)
# Drop all missing values in 'main_race'
all_years = all_years.dropna(subset = ["main_race"])
```

[20]:
```
all_years.head()
```

[20]:

	stop_id	stop_cause	service_area	subject_race	subject_sex	\
0	1044975	Moving Violation	110	W	M	
1	1044976	Moving Violation	320	W	M	
2	1044977	Moving Violation	320	L	M	
3	1044978	Moving Violation	610	W	M	
4	1044980	Equipment Violation	930	H	M	

	subject_age	date_stop	time_stop	sd_resident	arrested	searched	day	month	\
0	24	2014-01-01	1:25	Y	N	N	01	01	
1	42	2014-01-01	5:47	Y	N	N	01	01	
2	29	2014-01-01	7:46	Y	N	N	01	01	
3	23	2014-01-01	8:10	Y	N	N	01	01	
4	35	2014-01-01	8:35	N	N	N	01	01	

	division	race	main_race
0	Northern	White	White
1	Eastern	White	White
2	Eastern	Laotian	Asian
3	Western	White	White
4	NorthWestern	Hispanic	Hispanic

1.11 Data Analysis & Results

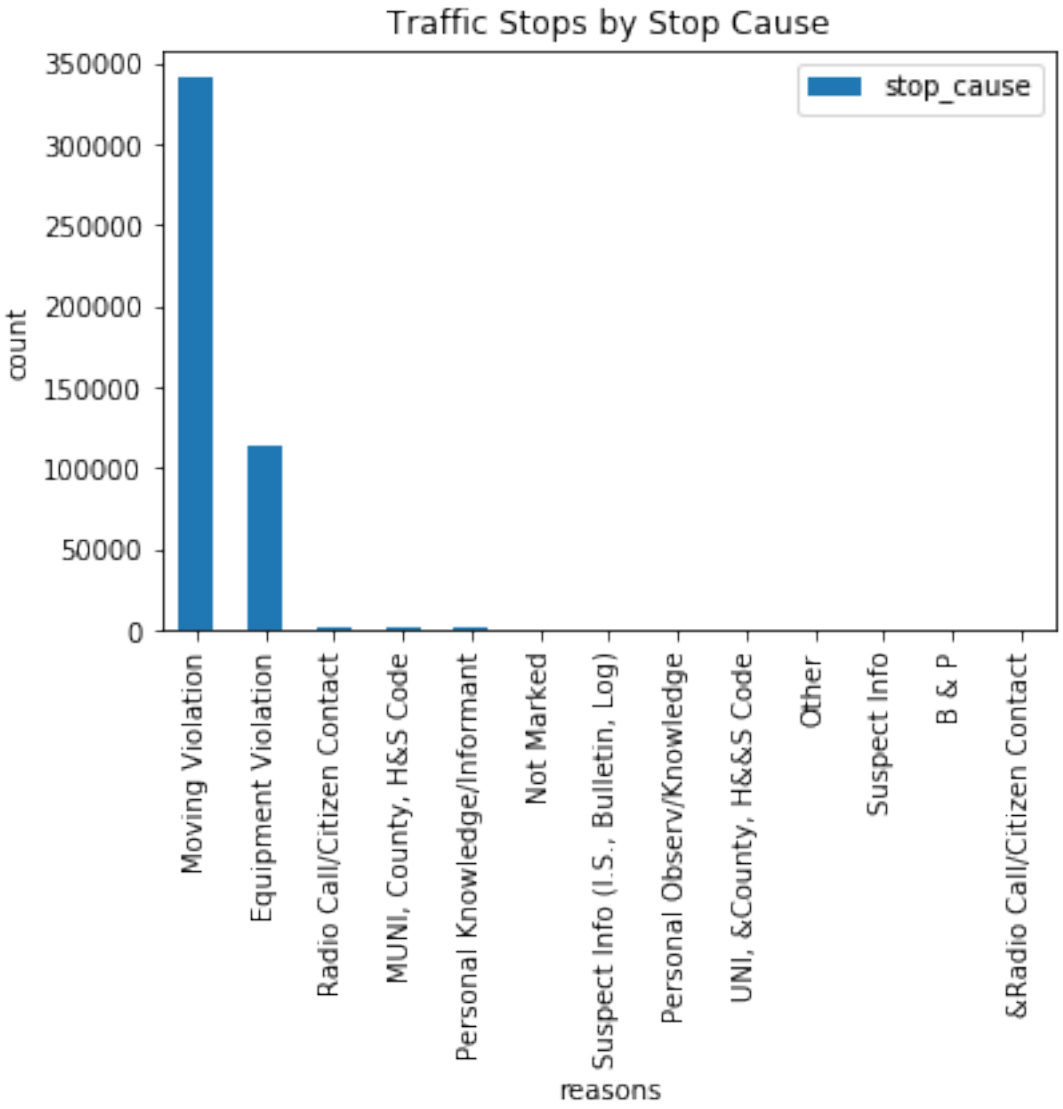
1.11.1 EDA

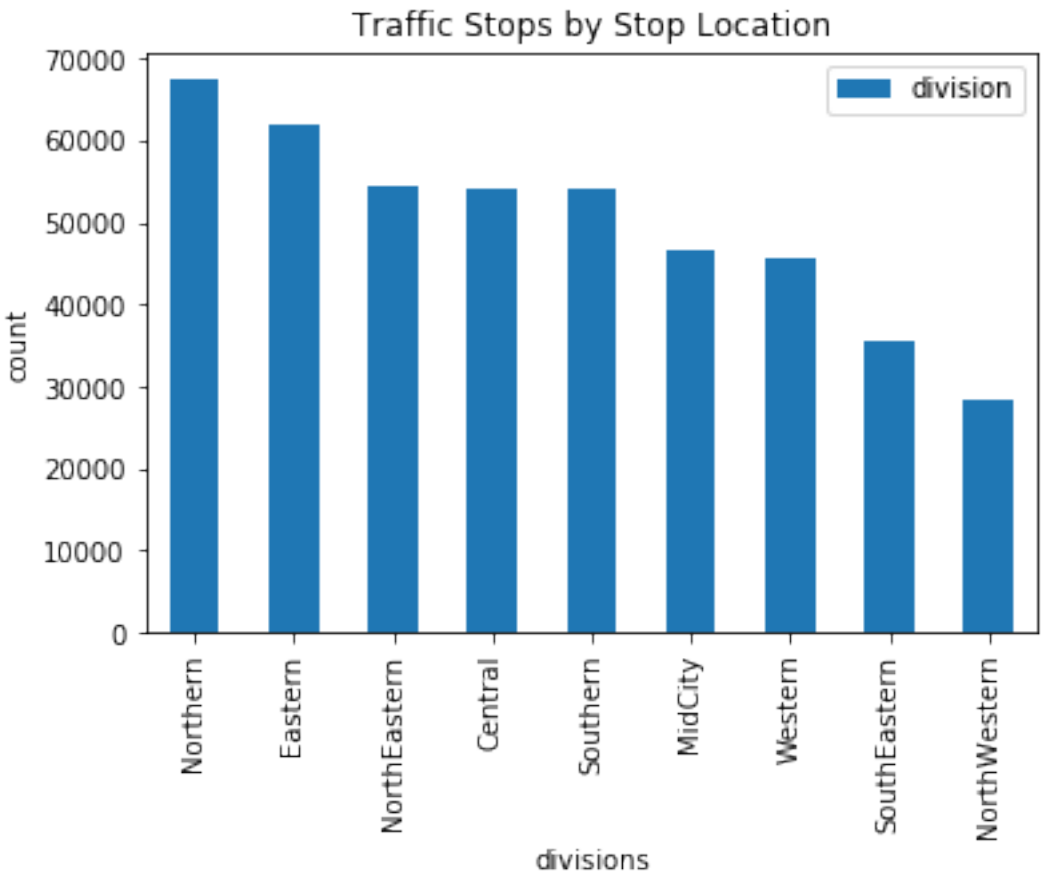
- Now that we have cleaned and standardized most of our data, let's do some exploratory data analysis.
- We will start by doing some visualizations for frequency of traffic stops for the variables which we would like to analyze.

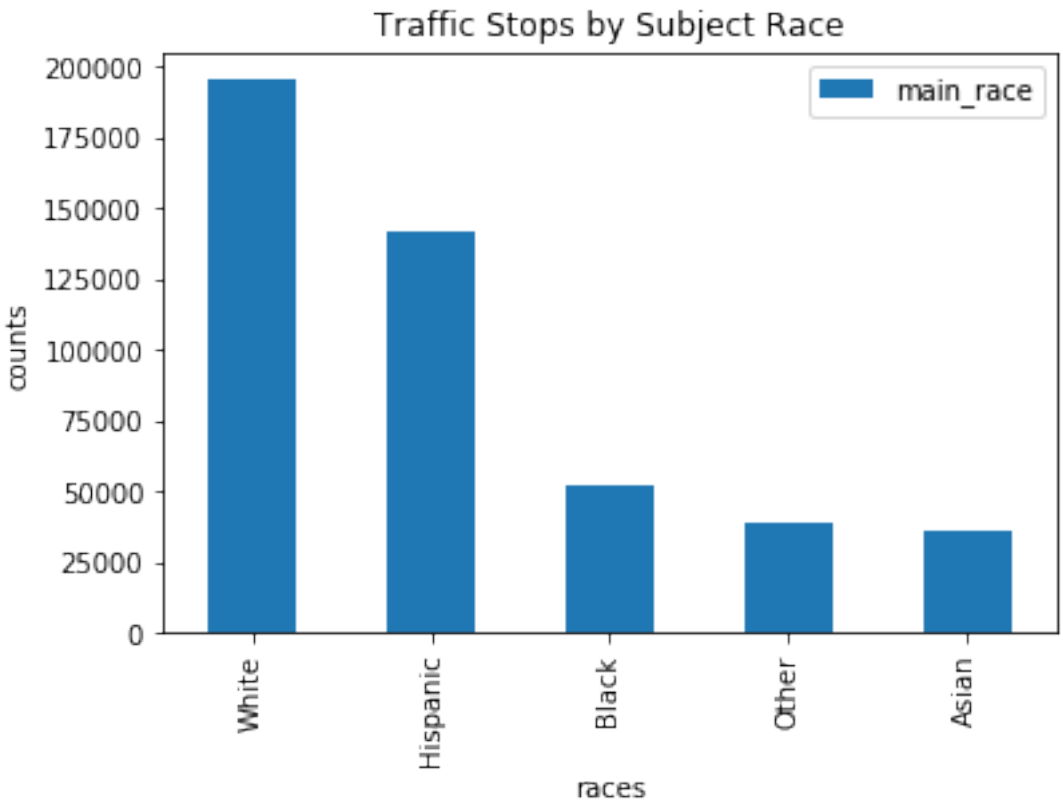
[21]:
```
stop_cause = all_years["stop_cause"].value_counts().to_frame()
stop_plot = stop_cause.plot(kind = "bar", title = "Traffic Stops by Stop Cause")
stop_plot.set_xlabel('reasons')
stop_plot.set_ylabel('count')

division = all_years["division"].value_counts().to_frame()
d = division.plot(kind = "bar", title = "Traffic Stops by Stop Location")
d.set_xlabel('divisions')
d.set_ylabel('count')

race = all_years["main_race"].value_counts().to_frame()
r = race.plot(kind = "bar", title = "Traffic Stops by Subject Race")
r.set_xlabel('races')
r.set_ylabel('counts')
```

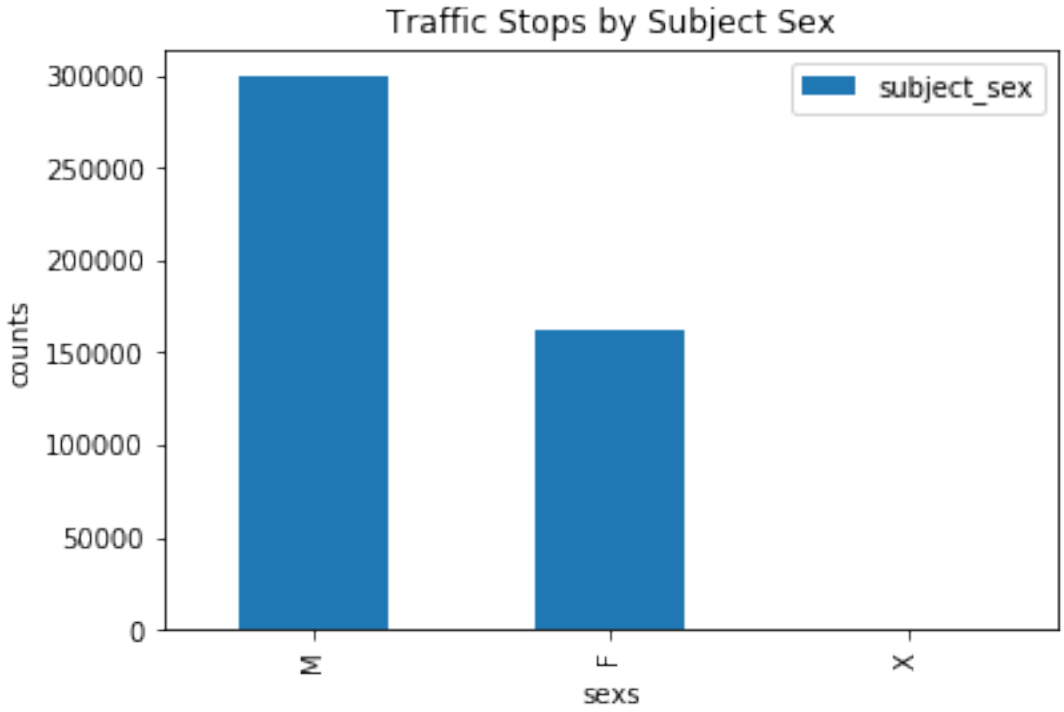
```
sex = all_years['subject_sex'].value_counts().to_frame()
s = sex.plot(kind= 'bar', title = 'Traffic Stops by Subject Sex')
s.set_xlabel('sexs')
s.set_ylabel('counts')
```

[21]: Text(0, 0.5, 'counts')

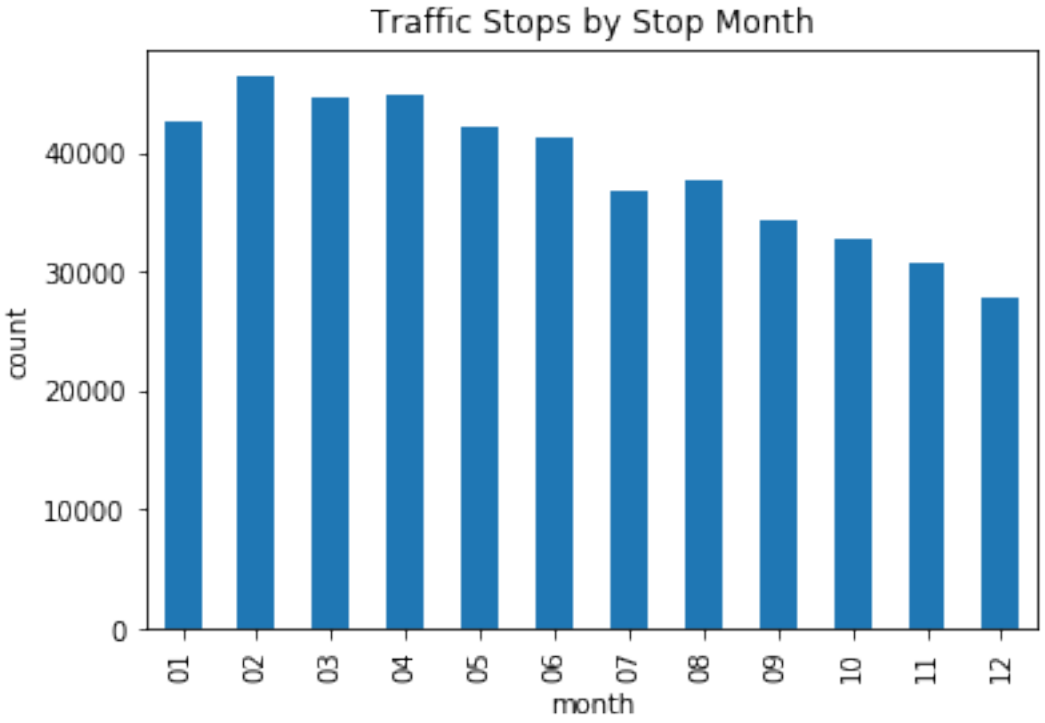


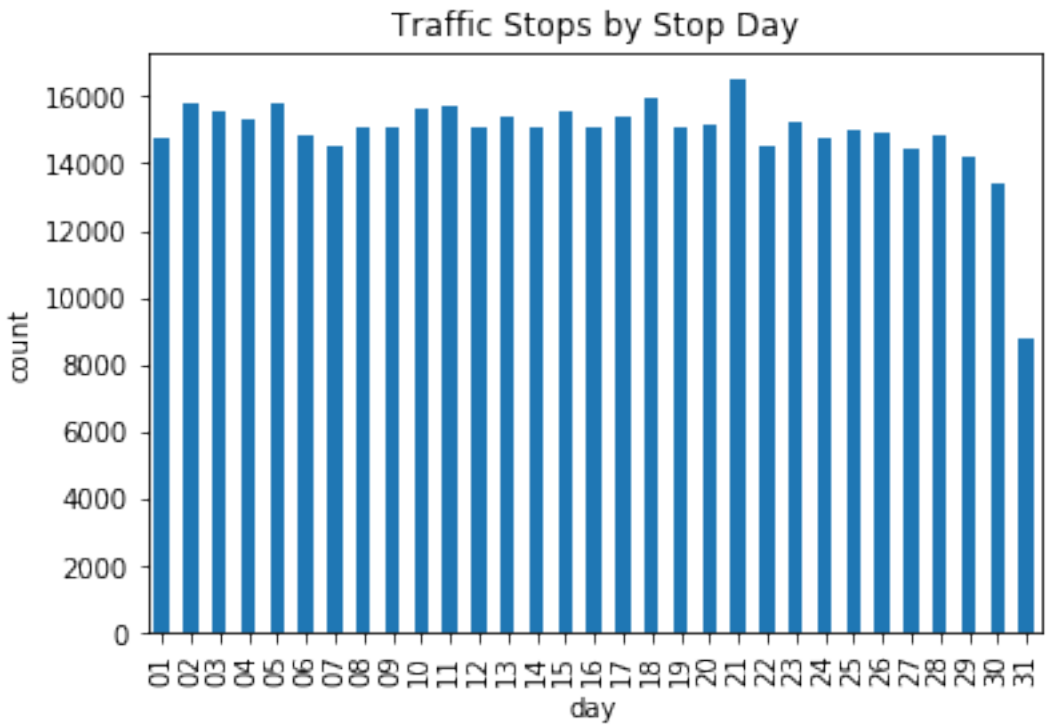

Traffic Stops by Stop Location
division
70000
60000
50000
40000
30000
20000
10000
0
count
Northern
Eastern
NorthEastern
Central
Southern
MidCity
Western
SouthEastern
NorthWestern
divisions

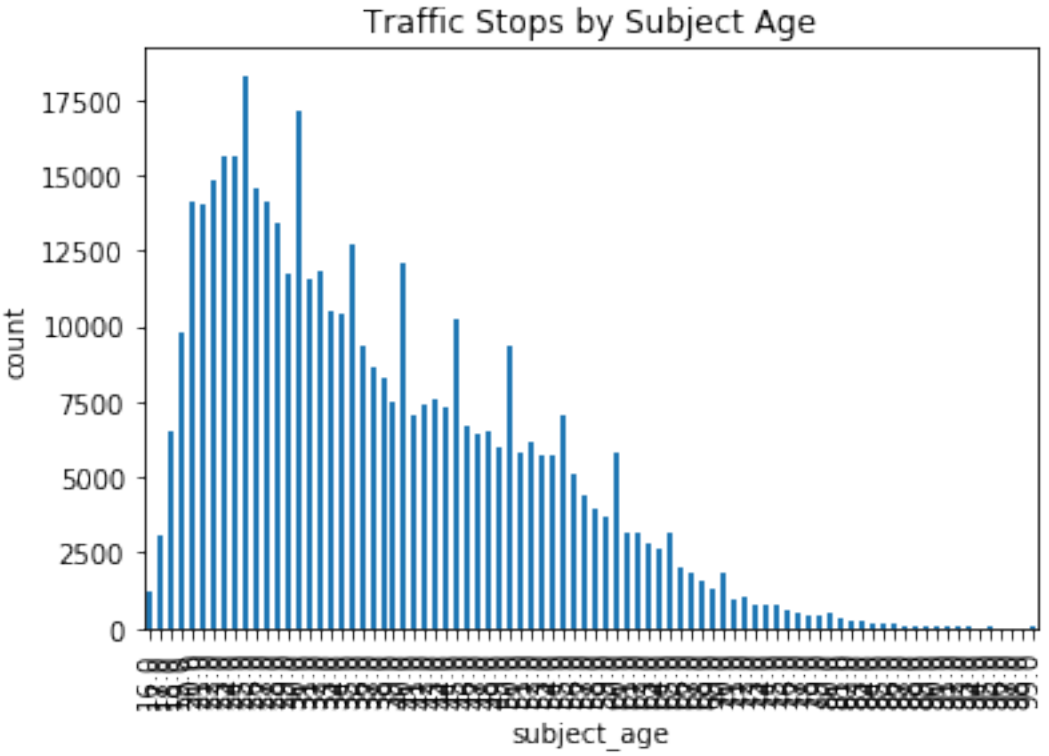

Traffic Stops by Subject Race
main_race
200000
175000
150000
125000
100000
75000
50000
25000
0
counts
White
Hispanic
Black
Other
Asian
races


Traffic Stops by Subject Sex
subject_sex
300000
250000
200000
150000
100000
50000
0
counts
M
F
X
sexs

[22]:
```
month = (all_years["month"].groupby(all_years["month"]).count()).
 ↪plot(kind="bar", title = "Traffic Stops by Stop Month")
plt.ylabel('count')
plt.show()
day = (all_years["day"].groupby(all_years["day"]).count()).plot(kind="bar",␣
 ↪title = "Traffic Stops by Stop Day")
plt.ylabel('count')
plt.show()
age = (all_years["subject_age"].groupby(all_years["subject_age"]).count()).
 ↪plot(kind="bar", title = "Traffic Stops by Subject Age")
plt.ylabel('count')
plt.show()
```




Traffic Stops by Stop Day
count
day
0
2000
4000
6000
8000
10000
12000
14000
16000
01
02
03
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31


Traffic Stops by Subject Age
count
subject_age
0
2500
5000
7500
10000
12500
15000
17500

- Examining our results above, we see that the graphs for Traffic Stops by Stop Day and Traffic Stops by Stop Month are relatively uniform so it doesn't seem as though they should be further analyzed.
- Taking a look at the Traffic Stops by Stop Cause graph, we see that the causes are only due to moving violations and equipment violations. These are logical and also don't require further analysis.

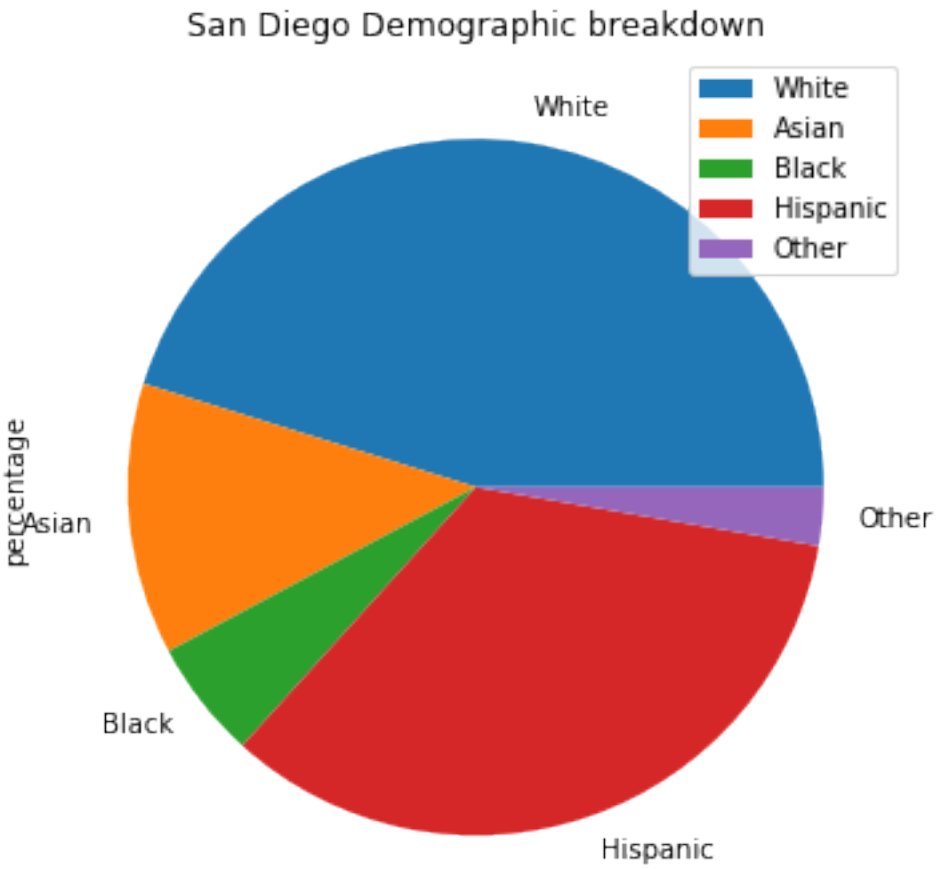
- For the graph on Subject Race, we can see that the subject race which is pulled over most often is white, hispanic, and then black. At first glance, there is a skewed, unimodal distribution with whites subjected to traffic stops at higher rates than all other demographics, police officers are more sensitive to white individuals. Before we jump to any conclusions, we should first take a look at the population's demographic makeup.
- For the histogram on subject age, we can easily see that there is a skewed distribution with young people being pulled over at a much higher rate than their older counterparts. It would be interesting to dip into this data further and really look to see if there is any greater relationship.

[23]:
```
clean_df = all_years.
 ↪drop(columns=['stop_id','stop_cause','service_area','sd_resident','arrested','searched','ra
clean_df.head()
```

[23]:

	subject_sex	subject_age	division	main_race
0	M	24	Northern	White
1	M	42	Eastern	White
2	M	29	Eastern	Asian
3	M	23	Western	White
4	M	35	NorthWestern	Hispanic

[24]:
```
race_percentage = {"White": 45.2, "Asian": 12.6, "Black": 5.5, "Hispanic": 34.
 ↪0, "Other": 2.7}
race_df = pd.DataFrame({'percentage': [45.2, 12.6, 5.5, 34.0, 2.7]},
                  index=["White","Asian" ,"Black", "Hispanic", "Other"])
race_plot = race_df.plot.pie(y='percentage', figsize=(6, 6),title="San Diego␣
 ↪Demographic breakdown")
```



The pie chart above provides a visualization of the makeup of San Diego county. We can see that the largest demographic consists of white individuals which account for about half of the population. What's interesting is that Black people only contribute to about 5% of the population, and yet have been pulled over a little more than 50,000 times in the last four years. Let's perform some further analysis by normalizing the data.

1.11.2 Analysis

- what analytical approaches you are going to do (ie.expected vs actual and what anaytical approaches you are going to use)

[25]:
```
def normalize(makeup):
    total = len(all_years)
    return int((makeup/100) * total)
```

[26]:
```
total = len(all_years)
temp = []
```

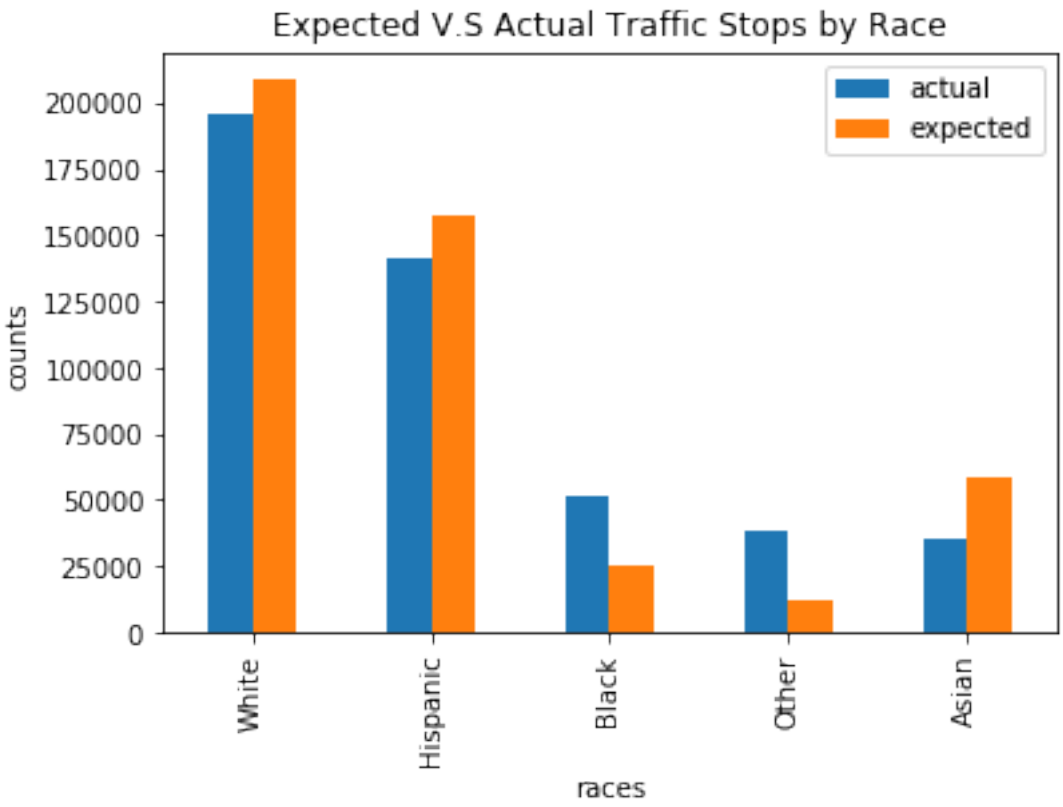
```
for i in race_percentage:
    temp.append(race_percentage[i])
race["makeup"] = race.index
race["makeup"].replace({"White": race_percentage["White"], "Hispanic":␣
↪race_percentage["Hispanic"], "Other": race_percentage["Other"],"Asian":␣
↪race_percentage["Asian"],"Black": race_percentage["Black"]}, inplace=True)

race["expected"] = race["makeup"].apply(normalize)
race.rename(columns={"main_race":"actual"}, inplace=True)
race
```

[26]:

	actual	makeup	expected
White	195324	45.2	208956
Hispanic	141188	34.0	157179
Black	51564	5.5	25426
Other	38595	2.7	12481
Asian	35623	12.6	58249

[27]:
```
r = race[["actual", "expected"]].plot(kind = "bar", title = "Expected V.S␣
↪Actual Traffic Stops by Race")
r.set_xlabel('races')
r.set_ylabel('counts')
```

[27]:
```
Text(0, 0.5, 'counts')
```



The expected values for White and Hispanic individuals are quite similar to their actual values while Black, Other and Asians are quite far from their actual scores. In order to further analyze the data in a quantitative manner, let's perfrom a chi-squared test to check for significance.

[28]:
```
#chi squared test T
stats.chisquare(race.actual.values, race.expected.values)
```

[28]:
```
Power_divergenceResult(statistic=92813.2348783202, pvalue=0.0)
```

This is below 0.01 and shows that it is statistically significant. This means there's a very low chance that we'd see this result if the police were pulling over subjects evenly across all races. From the chi square test we can see that race plays a role in who the police pulls over for traffic stops. This aligns with previous findings that found that the San Diego police department were being discriminatory. We will also look at categories such as sex and age. From the plots drawn above it seems like males are getting pulled over at a disproportionate rate then females and that younger people are being pulled over a lot more than older people.

[29]:
```
def grouped_age(age):
    if age < 35:
        age = 'young'
    elif age < 55:
        age = 'middle'
```

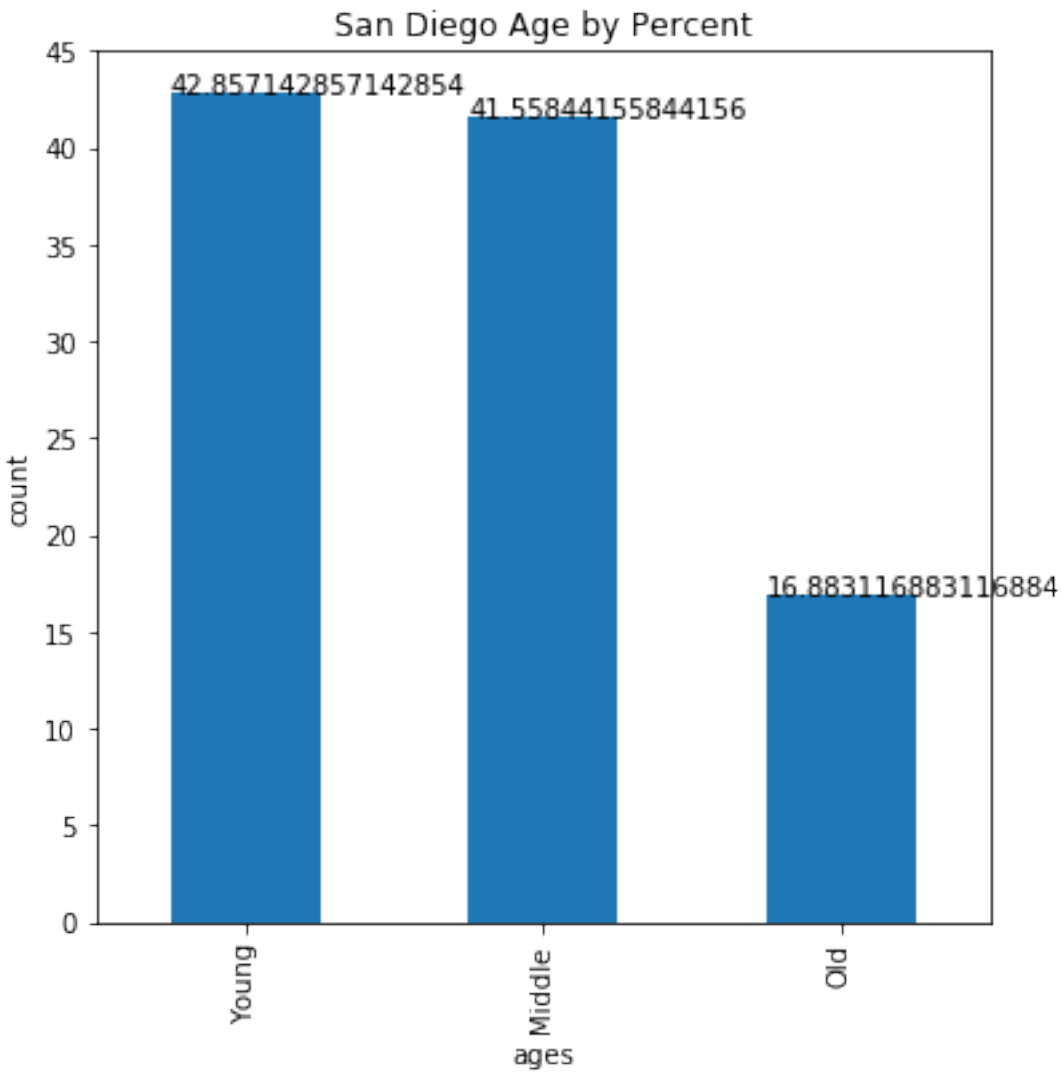
```
    else:
        age = 'old'
    return age
```

[30]:
```
clean_df = clean_df.dropna()
clean_df['subject_age'] = clean_df['subject_age'].astype(int)
clean_df["grouped_age"] = clean_df['subject_age'].apply(grouped_age)
```

[31]:
```
young = (7+9+17)/77*100
middle = (16+12+4)/77*100
old = (3+5+4+1)/77*100
len_young = len(clean_df[clean_df['grouped_age']=='young'])
len_middle = len(clean_df[clean_df['grouped_age']=='middle'])
len_old = len(clean_df[clean_df['grouped_age']=='old'])
```

[32]:
```
##### age_percentage = {'young':young, "middle":middle, 'old':old}
age_df = pd.DataFrame({'percentage': [young,middle,old]},
                      index=['Young','Middle','Old'])
age_plot = age_df.plot.bar(y='percentage', figsize=(6, 6),title="San Diego Age␣
 ↪by Percent",legend=False)
for p in age_plot.patches:
    age_plot.annotate(str(p.get_height()), xy=(p.get_x(), p.get_height()))
age_plot.set_xlabel('ages')
age_plot.set_ylabel('count')
```

[32]:
```
Text(0, 0.5, 'count')
```



This shows the young and middle aged people are pulled over much more than older people in San Diego. Note that the young age group is composed of people aged 16 to 34. Middle aged runs from 35 to 54, and the old age group is 55 and over.

[33]:
```
total = len(all_years)
temp = []
age_percentage = {'Young':young, 'Middle':middle, "Old":old}
for i in age_percentage:
    temp.append(age_percentage[i])
age_df['actual'] = [len_young,len_middle,len_old]
age_df["makeup"] = age_df.index
age_df['makeup'] = [42.85,41.55,16.8]
age_df["expected"] = age_df["makeup"].apply(normalize)
#age_df.rename(columns={"grouped_age":"actual"}, inplace=True)
```

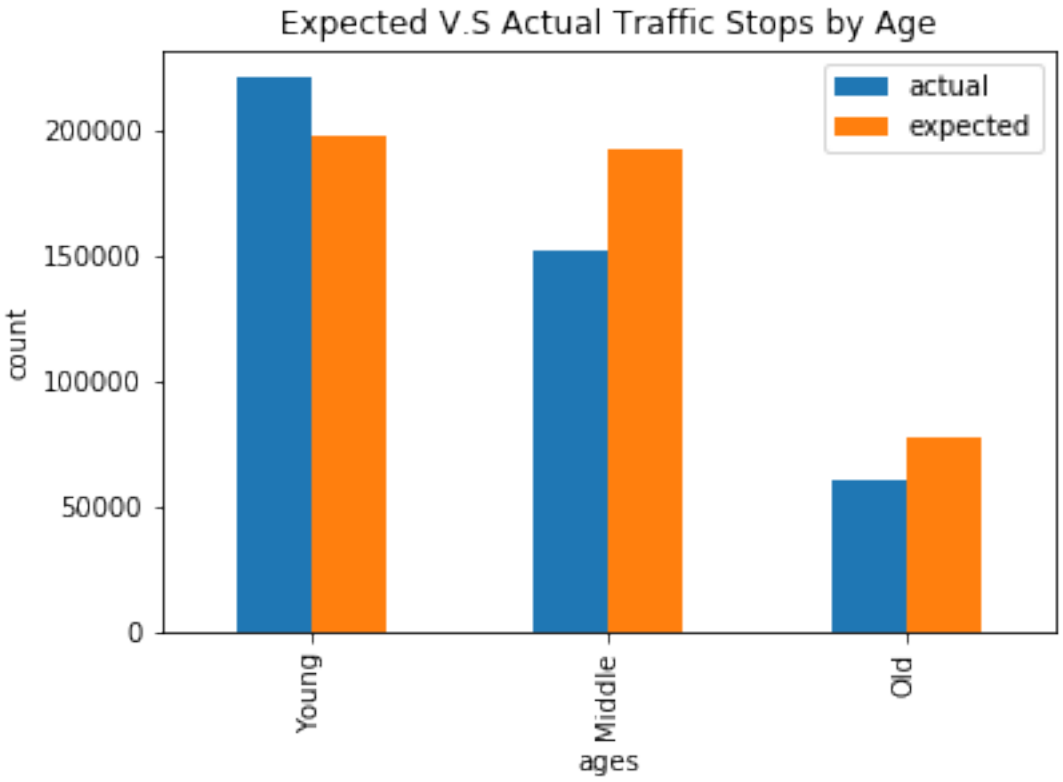
```
age_df
```

[33]:

	percentage	actual	makeup	expected
Young	42.857143	220719	42.85	198092
Middle	41.558442	152165	41.55	192083
Old	16.883117	60346	16.80	77665

[34]:
```
age = age_df[["actual", "expected"]].plot(kind = "bar", title = "Expected V.S␣
↪Actual Traffic Stops by Age")
age.set_xlabel('ages')
age.set_ylabel('count')
```

[34]:
```
Text(0, 0.5, 'count')
```



[35]:
```
stats.chisquare(age_df.actual.values, age_df.expected.values)
```

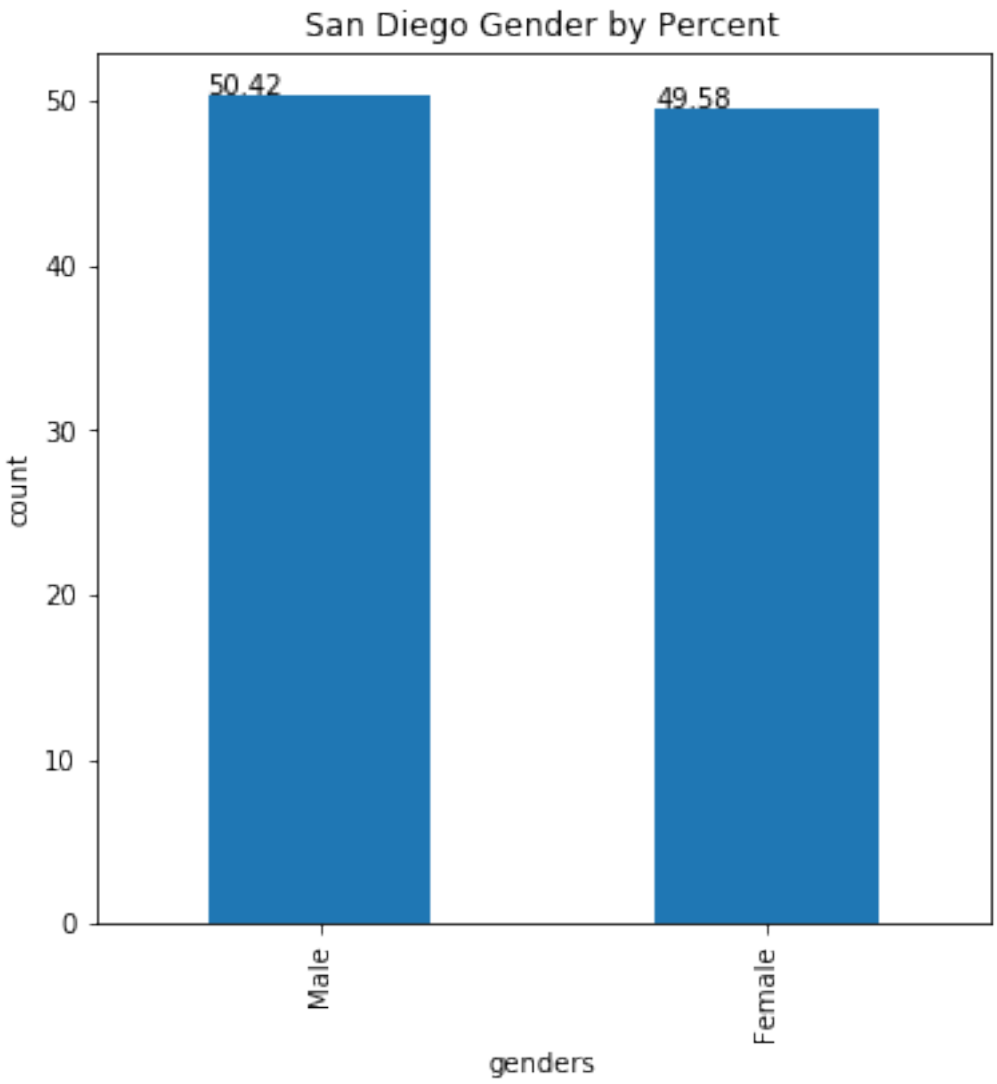
[35]:
```
Power_divergenceResult(statistic=14742.24914815308, pvalue=0.0)
```

This is below 0.01 and shows that it is statistically significant. This means there's a very low chance that we'd see this result if the police were pulling over subjects evenly across all ages. From the chi square test we can see that age plays a role in who the police pulls over for traffic stops. This aligns with popular belief that younger drivers are more reckless and as a result get pulled over more.

From this we can conclues that younger people are more likely to get pulled over than people who are older.

[36]:
```
#plot San Diego gender proportion
Male = 50.42
Female = 49.58
sex_percentage = {'Male':50.42, "Female":49.58}
sex_df = pd.DataFrame({'percentage': [50.42,49.58]},
                  index=['Male','Female'])
sex_plot = sex_df.plot.bar(y='percentage', figsize=(6, 6),title="San Diego␣
↪Gender by Percent",legend=False)
for p in sex_plot.patches:
    sex_plot.annotate(str(p.get_height()), xy=(p.get_x(), p.get_height()))
sex_plot.set_xlabel('genders')
sex_plot.set_ylabel('count')
```

[36]:
```
Text(0, 0.5, 'count')
```

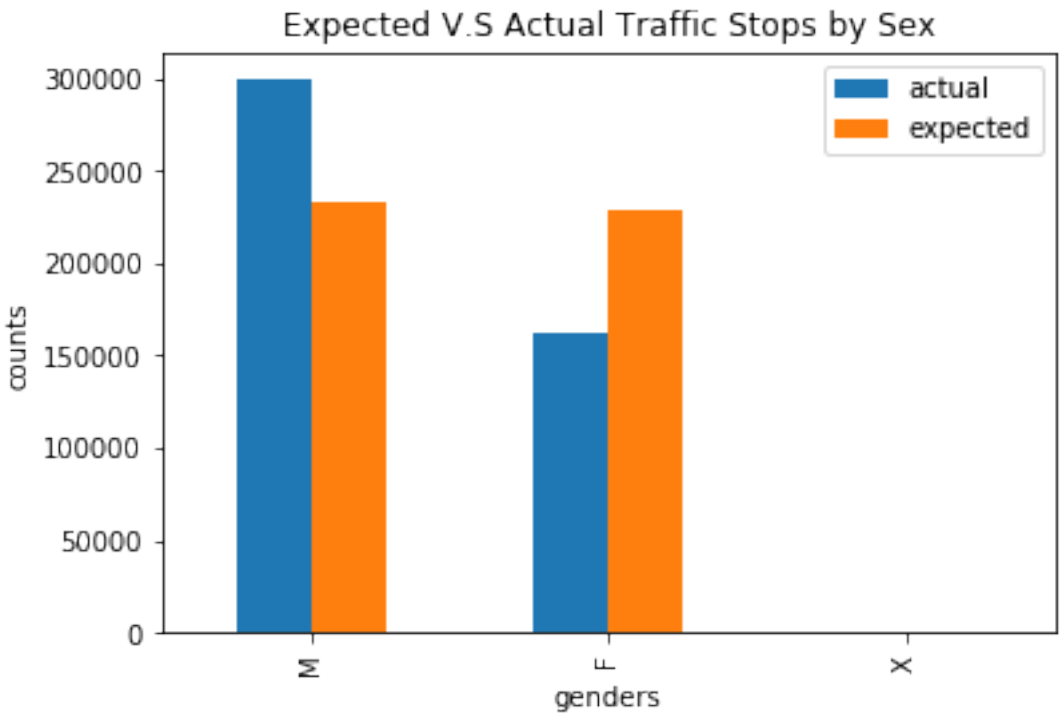


[37]:
```
#Create df of actual and expected counts of pull overs for sex
total = len(all_years)
temp = []
for i in sex_percentage:
    temp.append(sex_percentage[i])
sex["makeup"] = sex.index
sex["makeup"].replace({"M": 50.42, "F": 49.58, 'X':0.0}, inplace=True)
sex["expected"] = sex["makeup"].apply(normalize)
sex.rename(columns={"subject_sex":"actual"}, inplace=True)
```

[38]:
```
#plot expected vs actual for sex
ax = sex[["actual", "expected"]].plot(kind = "bar", title = "Expected V.S␣
↪Actual Traffic Stops by Sex")
```

```
ax.set_xlabel("genders")
ax.set_ylabel("counts")
```

[38]: `Text(0, 0.5, 'counts')`



[40]:
```
#chi square test for sex
stats.chisquare(sex.actual.values, sex.expected.values)
```

[40]: `Power_divergenceResult(statistic=inf, pvalue=0.0)`

This is below 0.01 and shows that it is statistically significant. This means there's a very low chance that we'd see this result if the police were pulling over subjects evenly across all genders. From the chi square test we can see that gender plays a role in who the police pulls over for traffic stops. This aligns with popular belief that males are more aggressive drivers and as a result are more likely to get pulled over.

Correlation

[41]:
```
#Drop na for age so we can graph
dropnans = all_years.dropna()
h_W = dropnans[dropnans['main_race'] == 'White']['subject_age'].values
h_B = dropnans[dropnans['main_race'] == 'Black']['subject_age'].values
```

[42]:
```
#Do the normal test for white and black race
st_W = normaltest(h_W)[0]
p_W = normaltest(h_W)[1]
st_B = normaltest(h_B)[0]
p_B = normaltest(h_B)[1]
```

[43]:
```
#Drop all of the null values so we can create a correlation matrix
clean_df = clean_df.dropna()
```

[44]:
```
#Hot encode the gender category
onehot = clean_df.replace('M',1)
onehot = onehot.replace('F',0)
```

[45]:
```
#Hot encode all of the string categories so we can create a correlation matrix
le_division = LabelEncoder()
le_main_race = LabelEncoder()
onehot['division_encoded'] = le_division.fit_transform(onehot.division)
onehot['main_race_encoded'] = le_main_race.fit_transform(onehot.main_race)
```

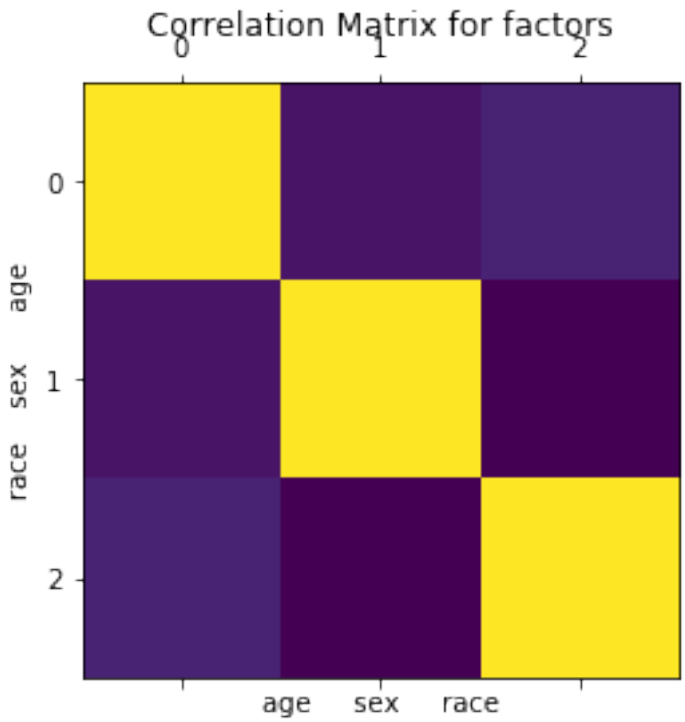
[46]:
```
#Drop all columns we are not investigating
onehot = onehot.drop(columns=['main_race','division'])
```

[47]:
```
#Create a correlation matrix
onehot.corr()
```

[47]:

	subject_age	division_encoded	main_race_encoded
subject_age	1.000000	0.030001	0.076579
division_encoded	0.030001	1.000000	-0.024054
main_race_encoded	0.076579	-0.024054	1.000000

[48]:
```
#plot the correlation matrix
plt.matshow(onehot.corr())
plt.title('Correlation Matrix for factors')
plt.ylabel('race    sex      age')
plt.xlabel('age     sex     race')
plt.show()
```



The matrix plot shows the correlations between each factors. It shows the correlation between sex, age, and race. From the chart and the picture above we can highly conclude that the variables are not correlated with one another.

Building Logit Model In order to use Logistic Regression on our data set we must first do post-stratification. Post-statification is "a method for adjusting the sampling weights, usually to account for underrep- resented groups in the population" (Wiki). This is because our data set only contains data of people that are pulled over. We had a missing data problem where it's missing not at random. As of this point we are only investigating three variables race, sex, and age. To do post-stratification we create a not pulled over dataset containing data that is representative of the actual population of San Diego. We assign this data to not pulled over and then concat this dataset to our original dataset.

[49]:
```
#Function to create race datafame
def pickrace():
    value = ['White','Asian','Black','Hispanic','Other']
    probabilites = [.452,.126,.055,.34,.027]
    race = np.random.choice(value,433230,p=probabilites)
    return race
```

[50]:
```
#Function to create sex datafame
def picksex():
    {'Male':50.42, "Female":49.58}
```

```
    value = ['M','F']
    probabilites = [.5042,.4958]
    sex = np.random.choice(value,433230,p=probabilites)
    return sex
```

[51]:
```
#Function to create age dataframe
def pickage():
    value = ['young','middle','old']
    probabilities = [.425,.413,.162]
    age = np.random.choice(value,433230,p=probabilities)
    return age
```

[52]:
```
race = pickrace()
sex = picksex()
age = pickage()
```

[53]:
```
race = pd.DataFrame(race,columns=['main_race'])
sex = pd.DataFrame(sex,columns=['subject_sex'])
age = pd.DataFrame(age,columns=['grouped_age'])
```

[54]:
```
#Create not pulled over dataframe
nulldf = pd.concat([sex,race,age],axis=1)
nulldf['pull_over'] = 0
```

[55]:
```
clean_df['pull_over'] = 1
cleandf = clean_df.drop(columns=['subject_age','division'])
```

[56]:
```
#Concat not pulled with original dataframe
model_df = pd.concat([cleandf,nulldf], axis=0)
model_df
```

[56]:

	subject_sex	main_race	grouped_age	pull_over
0	M	White	young	1
1	M	White	middle	1
2	M	Asian	young	1
3	M	White	young	1
4	M	Hispanic	middle	1
…	…	…	…	…
433225	M	Hispanic	young	0
433226	M	White	middle	0
433227	F	White	middle	0
433228	F	White	young	0
433229	M	Hispanic	middle	0

[866460 rows x 4 columns]

[57]:
```
cols=['subject_sex','main_race','grouped_age']
X=model_df[cols]
y=model_df['pull_over']
```

[58]:
```
#Change data so it can be put into the model. Use onehot encoding to transform␣
 ↪categorical data to numerical
le_sex = LabelEncoder()
le_age = LabelEncoder()
le_main_race = LabelEncoder()
model_df['sex_encoded'] = le_sex.fit_transform(model_df.subject_sex)
model_df['main_race_encoded'] = le_main_race.fit_transform(model_df.main_race)
model_df['group_age_encoded'] = le_age.fit_transform(model_df.grouped_age)

model_input = model_df.drop(columns=['subject_sex','main_race','grouped_age'])
model_input
```

[58]:

	pull_over	sex_encoded	main_race_encoded	group_age_encoded
0	1	1	4	2
1	1	1	4	0
2	1	1	0	2
3	1	1	4	2
4	1	1	2	0
…	…	…	…	…
433225	0	1	2	2
433226	0	1	4	0
433227	0	0	4	0
433228	0	0	4	2
433229	0	1	2	0

[866460 rows x 4 columns]

Test Train Split

[59]:
```
#Create a train and test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(model_input.
 ↪drop('pull_over',axis=1),
                                                    model_input['pull_over'],␣
 ↪test_size=0.20,
                                                    random_state=101)
```

Training and Predicting

[61]:
```
#Make predictions with our logistic regression model
from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression()
```

```
logmodel.fit(X_train,y_train)
predictions = logmodel.predict(X_test)
```

```
/opt/conda/lib/python3.6/site-packages/sklearn/linear_model/logistic.py:432:
FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify a
solver to silence this warning.
  FutureWarning)
```

[62]:
```
#Print result of our logistic model
from sklearn.metrics import classification_report
print(classification_report(y_test,predictions))
```

```
              precision    recall  f1-score   support

           0       0.58      0.50      0.54     86476
           1       0.56      0.65      0.60     86816

    accuracy                           0.57    173292
   macro avg       0.57      0.57      0.57    173292
weighted avg       0.57      0.57      0.57    173292
```

From our model we were only able to achieve a 58% accuracy on guessing whether someone was pulled over or not with logistic regression.

[63]:
```
#Create another logisitc regression model with statsmodels to find which␣
 ↪variable contributed the most
logit = sm.Logit(model_input['pull_over'],␣
 ↪model_input[['sex_encoded','main_race_encoded','group_age_encoded']])
result = logit.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.684786
         Iterations 4
```

[64]:
```
print(result.summary2())
```

```
                          Results: Logit
==================================================================
Model:              Logit            Pseudo R-squared: 0.012
Dependent Variable: pull_over        AIC:              1186686.0642
Date:               2020-03-19 12:29 BIC:              1186721.0807
No. Observations:   866460           Log-Likelihood:   -5.9334e+05
Df Model:           2                LL-Null:          -6.0058e+05
Df Residuals:       866457           LLR p-value:      0.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     4.0000
------------------------------------------------------------------
```

```
                    Coef.  Std.Err.     z     P>|z|    [0.025  0.975]
------------------------------------------------------------------------
sex_encoded         0.4113   0.0040 103.4797 0.0000   0.4035  0.4191
main_race_encoded  -0.0889   0.0011 -79.9005 0.0000  -0.0910 -0.0867
group_age_encoded   0.0698   0.0021  32.7274 0.0000   0.0656  0.0740
========================================================================
```

From this summary we are able to see that sex plays the most significant role in whether someone gets pulled over or not

[65]:
```
# Print out the corresponded factor of the maximum coefficient in the model␣
 ↪result
print(result.params.idxmax())
```

```
sex_encoded
```

[64]:
```
#We need to bootstrap our logistic regression model because we used␣
 ↪post-stratistication
variables = []
for i in range(100):
    model_input = model_input.sample(frac=1)
    logit = sm.Logit(model_input['pull_over'],␣
 ↪model_input[['sex_encoded','main_race_encoded','group_age_encoded']])
    result = logit.fit()
    variables.append(result.params.idxmax())
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

```
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
Optimization terminated successfully.
         Current function value: 0.684752
         Iterations 4
```

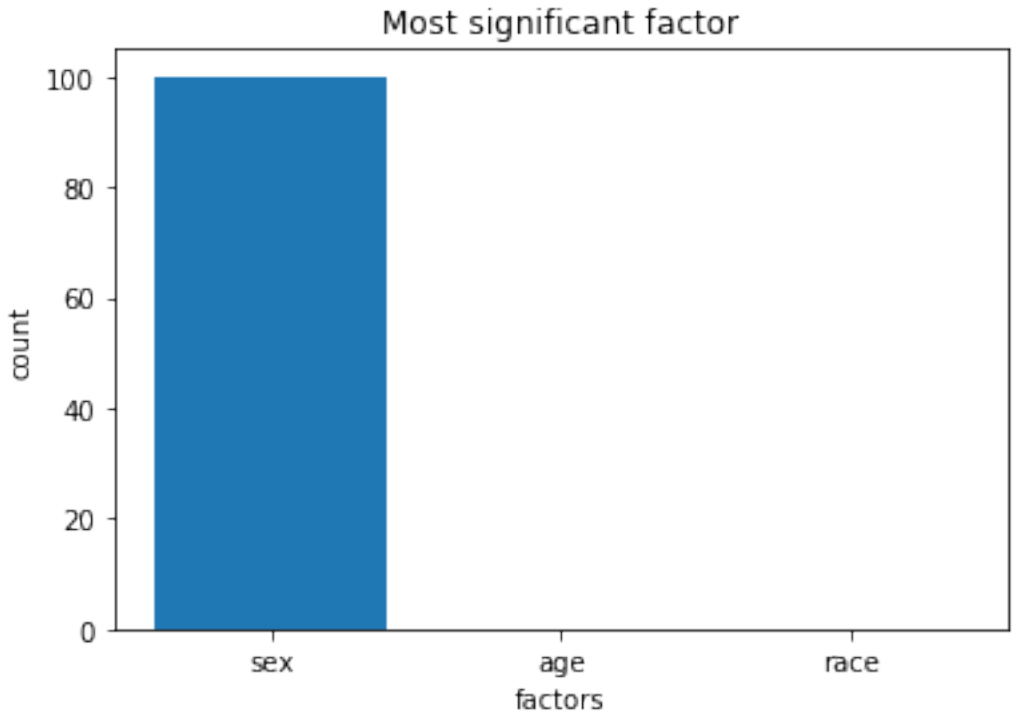
[65]:

```
#Out put the unique variables of the most significant variables
def unique(list1):
    x = np.array(list1)
    print(np.unique(x))

unique(variables)
```

```
['sex_encoded']
```

After using bootstrapping we can confirm that sex is indeed the most significant variable. We needed to do boot-strapping because we used post stratification.

[5]:
```
height = [100, 0, 0 ]
bars = ('sex', 'age', 'race')
y_pos = np.arange(len(bars))

# Create bars
plt.bar(y_pos, height)

plt.title('Most significant factor')

plt.xlabel('factors')
plt.ylabel('count')
# Create names on the x-axis
plt.xticks(y_pos, bars)

# Show graphic
plt.show()
```



This chart shows that out of the 100 bootstraps, sex was the most signifant factor in getting pulled over. Sex was the most significant factor in getting pulled over in every single iteration of the bootstrap.

1.12 Ethics and Privacy

In our research we considered the nine things that could ruin people's lives with data science. First off we considered our research question which was to find the most signifcant factor in pull overs. Our research question is ethical and well posed. We wanted to see if the police were discriminating against a certain subgroup of people or doing anything else unethical. Secondly we considered the implications of our research. Our research would affect the drivers and police department of San Diego and potentially the nation. From our research, police departments can try to address any potential discriminations helping the drivers of San Diego. One potential uninteded consequence could be riots because of public outrage from the findings (although this is very unlikely). We also considered the data we used. The data is publicly avaliable from the San Diego city website. The source is reliable and conforms to typical privacy concerns. The data is unable to trace back to anyone's identity. There does not seem to be any bias in our data also. Forth we considered informed consent. Informed consent does not apply to our research cause it does not have information about a single person, it is just pullover data. Fifth we considered privacy. Our data can not be traced back to a single person or police officer. Sixth we considered evaluation. The project would be considered successful if we found factors that contribute to getting pulled over. From this we will be able identify if there is any potential discriminations and as a result can help address it. Seventh we considered our analysis. Our analysis will be a correlation question. We will consider any confounding factors when conducting our research and make sure that it is accurate. Our research is also transparent. We will tell the people what techniques and models were used to find factors contributing to getting pulled over. Lastly we considered the long term effects of our research. We will continuously monitor the effects of the research question and see if the police traffic stops change over time.

As far as privacy concerns go, there aren't too many that aren't taken care of. The data we're using is readily available for us to use in our project, as it's public government data. All the identifying factors of any individual recorded in the data was also removed - age, sex, and race were recorded, but none of the personal identifiers listed in the Safe Harbour Method were recorded. As far as the privacy rights of those pulled over are concerned, it is unclear whether they themselves gave consent to their data being used like this - however, there isn't exactly anything we can do about that. We're also sure to keep in mind the overall socioeconomic traits of the area we're working with, and how the social climate of San Diego may affect the biases of those who collected the data.

Publishing this data could be problematic because analysis of the data can easily be used to support the direct opposite of what it implies. When comparing the statistics of police pull-overs along the lines of different races or different genders, the data only tells us "which race is pulled over more often", or "which gender is pulled over more often". This in particular would most likely point to a pattern in police behavior more than it would point to anything specific that those pulled over were doing. However, one could easily analyze this and use the data to implicitly support something the data does not say, such as "this race commits more crimes on the road than other races in the data". Needless to say, we don't really want this mistaken analysis to occur, and because of that, we'll be sure to provide a disclaimer or note that any conclusions drawn from this data don't provide any information about the causes of these stops. As an extra precaution, it might also be a good idea to not publicize this data.

1.13 Conclusion & Discussion

In conclusion our hypothesis was incorrect. We predicted that the greatest contributor to whether a person gets pulled over or not was their race. Contrary to our belief, from our data exploration and analysis we found that the greatest factor contributing to someone getting pulled over was a person's gender. We found that males are getting pulled over at a disproportionate rate compared to females. Even though males make up almost a perfect half of San Diego's population, they made up 65 percent of the pull overs in San Diego. After exploring all the variables we found that the race, gender, and age were significant variables. We found this by using the chi-square test. To conclude that gender was the most significant out of those three we created a logistic regression model and looked at the largest coefficient. We found that gender contributed the most to whether a person got pulled over or not.

This project had two major limitation that we ran into while conducting the analysis. One of the major limitation of the project was the dataset only included data of people who got pulled over. There was no data for people who did not get pulled over so we could not create any models based on the dataset we got. We had to get advice from Professor Ellis on how to impute data that was not missing at random. We came to the conclusion that we should do post-stratisfication and bootstraping to solve the issue. Another problem was the binary form of this data proved to be quite a challenge to work with. Instead of being able to perform more common data analysis techniques such as linear regression to the data we had to approach Professor Ellis and use her expertise to help us to understand how to perform a logistic regression. The binary nature means that apart from the histograms in the EDA section, it was especially difficult to include data visualization as a part of our data analysis.

This project has the potential to really aid the San Diego police force. While we found that gender was the largest factor for traffic stops, this shouldn't hide the fact that age and especially race were also significant variables. This project should be a wake up call to not just the San Diego police force, but to forces around the country to constantly be re-evaluating their techniques and practices to make sure they serve the citizens as best they can.

1.14 Team Contributions

- Overview (NICK)
- Research question (MATTHEW + NEVAN)
- Background & Prior Work (MATTHEW)
- Hypothesis, Datasets (MATTHEW + KELVIN + SHANNON)
- Data Cleaning (NEVAN + SHANNON + KELVIN + MATTHEW)
- Data Analysis: EDA (NEVAN + KELVIN), Analysis (NEVAN + KELVIN + SHANNON), Visualization (KELVIN + SHANNON)
- Ethics and Privacy (NICK)
- Conclusion (MATTHEW + NEVAN + KELVIN)