NYPD Civilian Complaints

This project contains data on 12,000 civilian complaints filed against New York City police officers. Interesting questions to consider include:

- Does the length that the complaint is open depend on ethnicity/age/gender?
- Are white-officer vs non-white complaintant cases more likely to go against the complainant?
- Are allegations more severe for cases in which the officer and complaintant are not the same ethnicity?
- Are the complaints of women more succesful than men (for the same allegations?)

There are a lot of questions that can be asked from this data, so be creative! You are not limited to the sample questions above.

Getting the Data

The data and its corresponding data dictionary is downloadable here.

Note: you don't need to provide any information to obtain the data. Just agree to the terms of use and click "submit."

Cleaning and EDA

- Clean the data.
  - Certain fields have "missing" data that isn't labeled as missing. For example, there are fields with the value "Unknown." Do some exploration to find those values and convert them to null values.
  - You may also want to combine the date columns to create a `datetime` column for time-series exploration.
- Understand the data in ways relevant to your question using univariate and bivariate analysis of the data as well as aggregations.

Assessment of Missingness

- Assess the missingness per the requirements in `project03.ipynb`

Hypothesis Test / Permutation Test

Find a hypothesis test or permutation test to perform. You can use the questions at the top of the notebook for inspiration.

Summary of Findings

Introduction

This project is aimed at exploring the complaints filed against NYPD officers and the details of the investigations from September 1985 to January 2020 provided by a dataset from New York City's Civilian Complain Review Board to find potential disparities in the time it took to resolve a case based on the demographics of a complaintant, officer, or other factors with the given details.

The dataset that I will be analyzing contains over 12,000 civilian complaints of NYC Police Officers with the board's investigation and details of the complaint cases such as: ethnicity, gender, and age of both the complaintant and officers. We're also given the dates of when commplaints were filed and when the cases closed, allowing for further analysis to see if a complaintant's demographics had an impact on the time it took to resolve their case.

I will be looking at the different ethnicities of the individuals(officers and complaintant) involved in a complaint case and will compare them to the time the case was open for in attempt to get insight on how ethnicities may affect an officer's response to an allegation case.
The question that I want to answer from my analysis is:
Does the ethnicity of the complaintant have an influence on the time length that their investigation case stays open?

Dataset Details:
Number of observations:

- 33358 rows, 26 columns
  Each observation representing an allegation and the 26 columns (unique_mos_id, first_name, month_received, month_closed, year_received, year_closed, complaintant_ethnicity, mos_ethnicity, etc.)

Cleaning and EDA

- First I loaded the 'allegations.csv' dataset into a Pandas dataframe named allegations, as well as a dataset 'nypd_col_names.csv', which provided the name of our columns and what they represent.
- Then, I dropped columns not relevant to my posed question.
- I then added a new column accordingly to my goal of finding durations.

Assessment of Missingness

TODO

Hypothesis Test

I found for black ethincity, the p-value was 0.0, showing that the null can be rejected. We can't assume the black complainants have longer case lengths though, because many factors also play into it, such as the number of cases.

Code

In [138…

```
import matplotlib.pyplot as plt
import numpy as np
import os
import pandas as pd
import seaborn as sns
%matplotlib inline
%config InlineBackend.figure_format = 'retina'  # Higher resolution figures
```

Cleaning and EDA

Data Cleaning

The dataset contains many columns that are not relevant to answering my posed question, so I will determine which will be removed. Along with dropping unnecessary columns, I need to add new information to evaluate time lengths of cases to analyze.

In [154…

```
# load dataset and column names
allegations_fp = os.path.join('data', 'allegations.csv')
allegations = pd.read_csv(allegations_fp)
display(allegations.head())

# dataset col names  to refer to
cols_fp = os.path.join('data', 'nypd_col_names.csv')
col_names = pd.read_csv(cols_fp)
# drop unnecessary columns
col_names = col_names[['field name', 'description']]
col_names = col_names.style.set_table_attributes('style="font-size: 10px"')
display(col_names)
```

	unique_mos_id	first_name	last_name	command_now	shield_no	complaint_id	month_received
0	10004	Jonathan	Ruiz	078 PCT	8409	42835	7
1	10007	John	Sears	078 PCT	5952	24601	11
2	10007	John	Sears	078 PCT	5952	24601	11
3	10007	John	Sears	078 PCT	5952	26146	7
4	10009	Noemi	Sierra	078 PCT	24058	40253	8

5 rows × 27 columns

	field name	description
0	unique_mos_id	unique ID of the officer ("member of service")

	field name	description
1	first_name	Officer's first name
2	last_name	Officer's last name
3	command_now	Officer's command assignment as of July 2020
4	complaint_id	Unique ID of the complaint
5	month_received	Month the complaint was received by CCRB
6	year_received	Year the complaint was received by CCRB
7	month_closed	Month the complaint investigation was closed by CCRB
8	year_closed	Year the complaint investigation was closed by CCRB
9	command_at_incident	Officer's command assignment at the time of the incident
10	rank_abbrev_incident	Officer's rank at the time of the incident, abbreviation
11	rank_abbrev_now	Officer's rank as of July 2020, abbreviation
12	rank_now	Officer's rank as of July 2020
13	rank_incident	Officer's rank at the time of the incident
14	mos_ethnicity	Officer's ethnicity
15	mos_gender	Officer's gender
16	mos_age_incident	Officer's age at time of incident
17	complainant_ethnicity	Complainant's ethnicity
18	complainant_gender	Complainant's gender
19	complainant_age_incident	Complainant's age at time of incident
20	fado_type	Top-level category of complaint
21	allegation	Specific category of complaint
22	precinct	Precinct associated with the complaint
23	contact_reason	Reason officer made contact with complainant
24	outcome_description	Outcome of the contact between office and complainant
25	board_disposition	Finding by the CCRB

In [155…

```
allegations = allegations.drop(['precinct', "rank_abbrev_incident", "rank_abbrev
```

In [156…

```
# checking the new dataframe
allegations.head()
```

Out[156…

	complaint_id	month_received	year_received	month_closed	year_closed	mos_ethnicity	mos_ge
0	42835	7	2019	5	2020	Hispanic	
1	24601	11	2011	8	2012	White	
2	24601	11	2011	8	2012	White	
3	26146	7	2012	9	2013	White	
4	40253	8	2018	2	2019	Hispanic	

Next, I want to calculate the time that a case has been open per complaint by combining the corresponding months and year and changing that date to a datetime object.

In [157…

```
def start_date(row):
    return str(row['month_received']) + '/' + str(row['year_received'])

def close_date(row):
    return str(row['month_closed']) + '/' + str(row['year_closed'])
```

Now, I want to add a new column to the allegations dataframe that represents the amount of time a case was open for called, "*duration*". The data in this column are the results from calculating the difference between the received and closed dates in months.

In [158…

```
def difference(dates):
    return (dates[1].year - dates[0].year) * 12 + dates[1].month - dates[0].mont

date1 = pd.to_datetime(allegations.apply(start_date, axis = 1))
date2 = pd.to_datetime(allegations.apply(close_date, axis = 1))

temp = pd.DataFrame([date1, date2]).T
time_diff = temp.apply(difference, axis = 1)

allegations = allegations.assign(date_received = date1)
allegations = allegations.assign(date_closed = date2)
allegations = allegations.assign(**{'duration' : time_diff})
```

I can now drop unneeded columns.

In [159…

```
allegations = allegations.drop(['month_received', 'year_received', 'month_closed
```

In [160…

```
allegations.head()
```

Out[160…

	complaint_id	mos_ethnicity	mos_gender	mos_age_incident	complainant_ethnicity	complainant
0	42835	Hispanic	M	32	Black	
1	24601	White	M	24	Black	
2	24601	White	M	24	Black	
3	26146	White	M	25	Black	
4	40253	Hispanic	F	39	NaN	

Here, we look for the columns in our modified dataset to see which of the values are unkown, as some of the missing values in the dataset were stored as 'Unknown'. I found the column

complaintant_ethnicity to be the only that contained "unknown" values. So the change function is applied to the given column.

In [161…
```
def unknown_to_nan(val):
    if val == 'Unknown':
        return np.NaN
    else:
        return val

allegations['complainant_ethnicity'] = allegations['complainant_ethnicity'].appl
```

In [162…
```
# checking cleaned df
allegations.head()
```

Out[162…

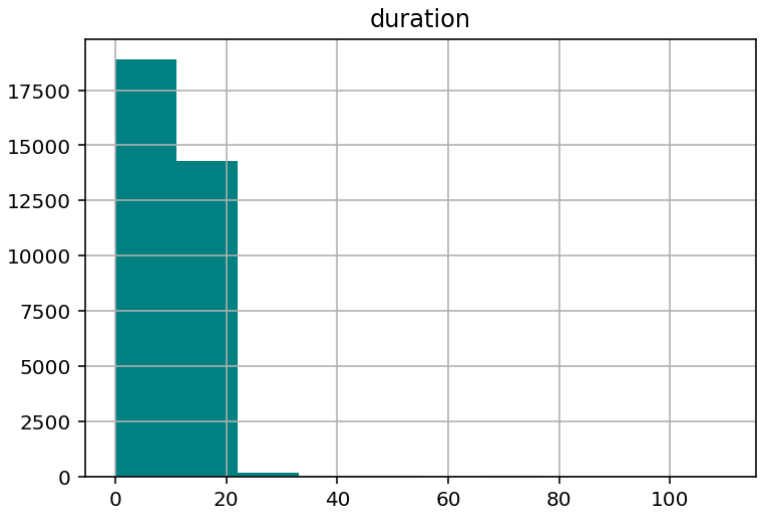
	complaint_id	mos_ethnicity	mos_gender	mos_age_incident	complainant_ethnicity	complainant
0	42835	Hispanic	M	32	Black	
1	24601	White	M	24	Black	
2	24601	White	M	24	Black	
3	26146	White	M	25	Black	
4	40253	Hispanic	F	39	NaN	

Univariate Analysis

I will use a univariate analysis of the durations in a histogram to see the distribution of the times in months it took to resolve a complaint case.

In [163…
```
allegations.hist(bins=10, column = "duration", color = "teal")
```

Out[163…
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fe49addb048>]],
      dtype=object)
```



In [164…

```
# getting stats of complaintants
allegations['complaint_id'].value_counts().describe()
```

Out[164…

```
count    12056.000000
mean         2.766921
std          2.266578
min          1.000000
25%          1.000000
50%          2.000000
75%          4.000000
max         30.000000
Name: complaint_id, dtype: float64
```

Below is the shortest, longest, and average duration of the cases.

In [165…

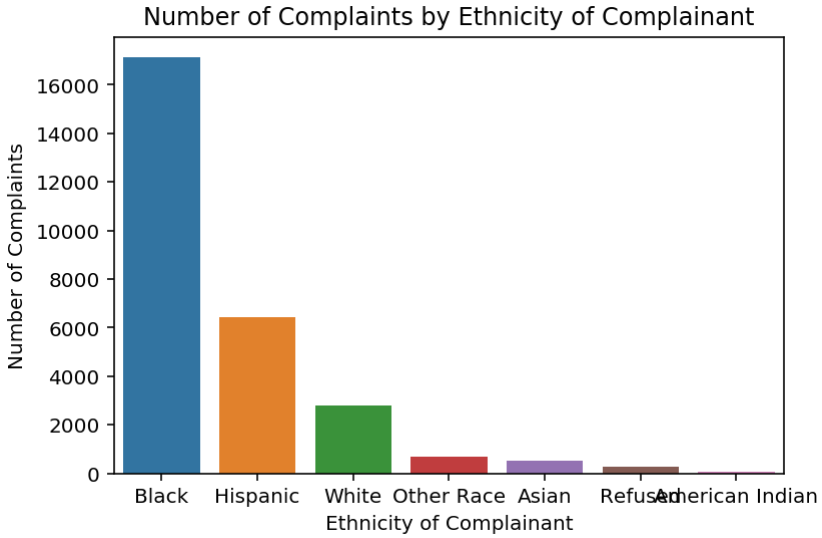
```
print('Shortest open time: ' + str(allegations['duration'].min()))
print('Longest open time: ' + str(allegations['duration'].max()))
print('Average open time: ' + str(allegations['duration'].mean()))
```

```
Shortest open time: 0
Longest open time: 110
Average open time: 9.733767012410816
```

Below is a plot of the number of complaints by the complaintant_ethnicity.

In [166…

```
# plotting the number of complaints based on complainant_ethnicity

new_df = allegations['complainant_ethnicity'].value_counts()
sns.barplot(x = new_df.index , y = new_df.values)
plt.title('Number of Complaints by Ethnicity of Complainant')
plt.xlabel('Ethnicity of Complainant')
plt.ylabel('Number of Complaints')
plt.show()
```

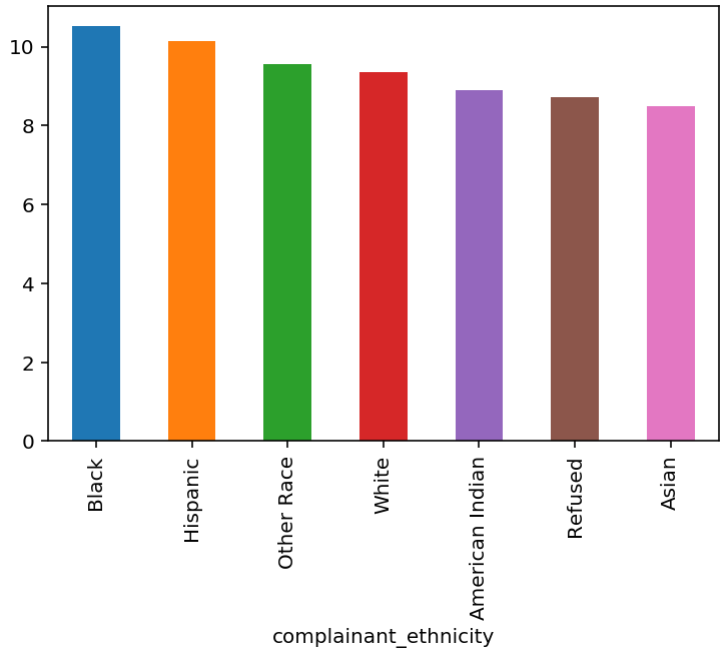


In [167…

```
df = allegations.groupby("complainant_ethnicity")["duration"].mean().sort_values
print(df)
df.plot.bar()
```

```
complainant_ethnicity
Black               10.515017
Hispanic            10.150218
Other Race           9.565731
White                9.359684
American Indian      8.906250
Refused              8.729730
Asian                8.488722
Name: duration, dtype: float64
```

Out[167… <matplotlib.axes._subplots.AxesSubplot at 0x7fe4835d64e0>



I notice a disparity in the time it took on a Black complaintants' complaint. Though, we cannot assume that officers took longer time to handle these cases, as there could've been higher rates of cases by Black complaintants.

Bivariate Analysis

In [168…

```
alleg_copy = allegations.copy()
plt.figure(figsize = (8,8))
sns.relplot(x ='date_received' , y = 'duration', data = alleg_copy, kind = 'line
plt.title('Date of Complaint Received vs Time Open of Case')
plt.xlabel('Date of Complaint Received')
plt.ylabel('duration')
plt.show();
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-168-c935024ece15> in <module>()
      1 alleg_copy = allegations.copy()
      2 plt.figure(figsize = (8,8))
----> 3 sns.relplot(x ='date_received' , y = 'duration', data = alleg_copy, kind
= 'line' )
      4 plt.title('Date of Complaint Received vs Time Open of Case')
      5 plt.xlabel('Date of Complaint Received')

AttributeError: module 'seaborn' has no attribute 'relplot'
<Figure size 576x576 with 0 Axes>
```

Here we see a slight longer duration in open cases for younger complaintants.

In [171…

```
alleg_copy = allegations.copy()
plt.figure(figsize = (8,8))
ax = sns.scatterplot(x ='complainant_age_incident' , y = 'duration', data = alle
plt.title('Scatter Plot of Complainent Age vs Duration of Complaint Case')
plt.xlabel('Complainent Age')
plt.ylabel('Time Open in Months')
plt.show();
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-171-4a65e1720270> in <module>()
      1 alleg_copy = allegations.copy()
      2 plt.figure(figsize = (8,8))
----> 3 ax = sns.scatterplot(x ='complainant_age_incident' , y = 'duration', dat
a = alleg_copy )
      4 plt.title('Scatter Plot of Complainent Age vs Duration of Complaint Cas
e')
      5 plt.xlabel('Complainent Age')

AttributeError: module 'seaborn' has no attribute 'scatterplot'
<Figure size 576x576 with 0 Axes>
```

Above, we can see that durations of an open case took a fall around 2016.

Below, I will compare the ethnicities of complainants against White officers, as we are trying to see if race plays a role in judicial responses and evaluating against a "dominant" group would help with insight.

In [ ]:

```
pivot = allegations.pivot_table(index = 'mos_ethnicity', columns ='complainant_e
                        aggfunc = 'count').fillna(0)
display(pivot)
plt.figure(figsize = (20,20))
pivot.plot(kind = 'bar')
plt.title('Count of Total Number of Complainant by Ethnicity for White Officers'
plt.xlabel('Ethnicity of Offcer')
```

```
plt.ylabel('Count of Complaints')
plt.show;
```

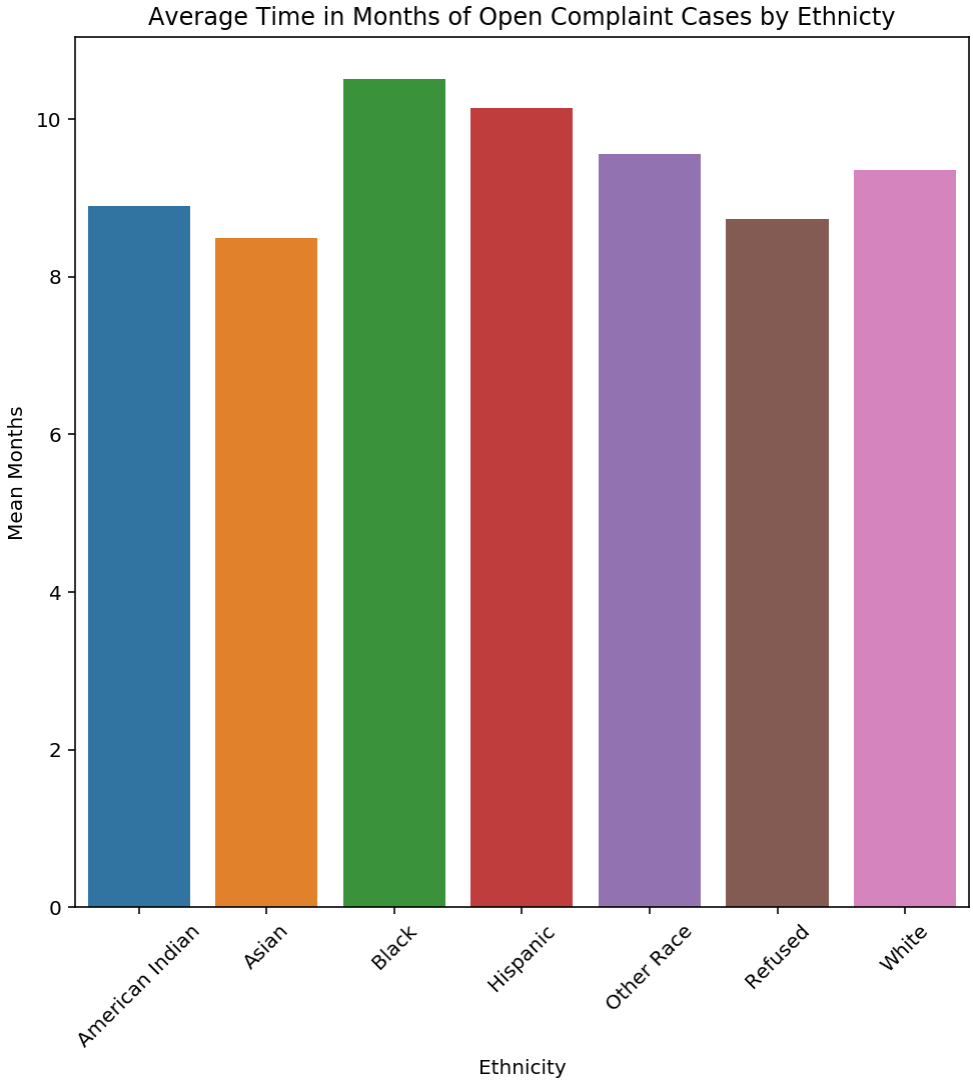
We can see a great difference between the complainant's ethnicity against white officers. However, we can see that black complainants are in large numbers against all officers, despite ethnicities.

Interesting Aggregates

Below is a chart comparing the average duration of an open case by ethnicity.

In [172…

```
ethnicity_grouped = allegations.groupby('complainant_ethnicity')['duration'].mea
plt.figure(figsize=(8,8))
ax = sns.barplot(x = ethnicity_grouped.index , y = ethnicity_grouped.values)
plt.title('Average Time in Months of Open Complaint Cases by Ethnicty')
plt.ylabel('Mean Months')
plt.xlabel('Ethnicity')
plt.xticks(rotation = 45)
plt.show();
```



Assessment of Missingness

I want to know whether the missingness of the complainant_gender is dependent on ethnicity. To do this, I need to use a permuation test to assess dependency.
I ended up choosing complaintant_ethnicity as my base column to test dependency on other columns, because there was a pattern in missing data whenever this column was missing.

I believe the data is NMAR because in intimidating circumstances, people may not be willing to give certain information about themselves to law officials.

Level of Significance: 0.5

In [175…

```
distr = (
    allegations
    .assign(is_null=allegations.complainant_gender.isnull())
    .pivot_table(index='is_null', columns='mos_gender', aggfunc='size')
    .apply(lambda x:x / x.sum(), axis=1)
)

# permutation test with mos_gender labels shuffled
n_repetitions = 500

tvds = []
for _ in range(n_repetitions):

    # shuffle the mos_gender column
    shuffled_col = (
        allegations['mos_gender']
        .sample(replace=False, frac=1)
        .reset_index(drop=True)
    )

    # table
    shuffled = (
        allegations
        .assign(**{
            'mos_gender': shuffled_col,
            'is_null': allegations['complainant_gender'].isnull()
        })
    )

    # compute the tvd
    shuffled = (
        shuffled
        .pivot_table(index='is_null', columns='mos_gender', aggfunc='size')
        .apply(lambda x:x / x.sum(), axis=1)
    )

    tvd = shuffled.diff().iloc[-1].abs().sum() / 2
    # add it to the list of results

    tvds.append(tvd)
obs = distr.diff().iloc[-1].abs().sum() / 2
pval = np.mean(tvds > obs)
```

In [176…

```
pval
```

Out[176… 0.002

With a low pvalue, we can reject the null and conclude that the missingness of complainant_gender is dependent on mos_gender.

Hypothesis Test

I wanted to find a relationship between ethnicities involved in an allegation and the officers investigating. I focused my analysis on the avergae length of the complaints duration to see if there were significant difference by ethnicity. Now I will run a hypothesis test for each ethnicity. Null: The mean total length of complaints s the same for white ethnicities vs the other ethnicities we are testing. (comparing to white ethnicity per patterns in societal treatments). Alternate: The mean total length of complaints is the less for white ethnicities vs the other ethnicities we are testing.

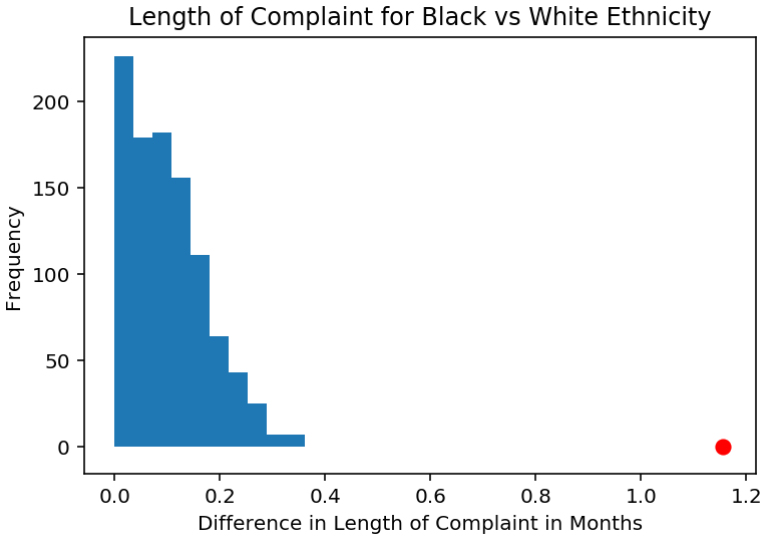
In [188…

```
def test_statistic(allegations, test_ethnicity, col):
    grouped = allegations.groupby('complainant_ethnicity')[col].mean()
    return abs(grouped['White'] - grouped[test_ethnicity])

def permutation_test(allegations, test_ethnicity):
    allegations_loced = allegations.loc[(allegations['complainant_ethnicity'] ==
    observed = test_statistic(allegations_loced, test_ethnicity, 'duration')
    shuffled_stats = []

    for i in range(1000):
        shuffled = allegations_loced['duration'].sample(frac=1, replace=False).r
        with_shuffled = allegations_loced[['complainant_ethnicity']].assign(shuf
        shuffled_stat = test_statistic(with_shuffled, test_ethnicity, 'shuffled'
        shuffled_stats.append(shuffled_stat)

    p_value = np.count_nonzero(shuffled_stats > observed)/float(1000)
    return p_value, shuffled_stats, observed
```

In [190…

```
p, stats, obs = permutation_test(allegations, 'Black')
pd.Series(stats).plot(kind='hist', title = 'Length of Complaint for Black vs Whi
plt.scatter(obs, 0, color='r', s=50)
plt.xlabel('Difference in Length of Complaint in Months')
print('p-value: '+ str(p))
print('obs: ' + str(obs))
```

```
p-value: 0.0
obs: 1.155333150724669
```


Length of Complaint for Black vs White Ethnicity
Frequency
200
150
100
50
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Difference in Length of Complaint in Months

In [ ]: