Homework 1 - Diagnostic

Due April 4, 2017

By Zhuo Leng

Problem 1: Data Acquisition and Analysis

In [5]:
```
import pandas as pd
from datetime import datetime, timedelta
import numpy as np
import matplotlib.pyplot as plt
```

1. You should download and combine data from the open data portal for the past year about the following 311 requests

(I choose not to combine the four dataset after I seeing the piazza)

In [6]:
```
##import data
graffiti = pd.read_csv('Graffiti_Removal.csv')

pot = pd.read_csv('Pot_Holes_reported.csv', names =['Creation Date', 'Sta
       'Type of Service Request', 'Current Activity',
       'Most Recent Action', 'Number of Potholes Filled On Block', 'Stree
       'X Coordinate', 'Y Coordinate', 'Ward', 'Police District',
       'Community Area', 'SSA', 'Latitude', 'Longitude', 'Location'], ski

sanitation = pd.read_csv('Sanitation_Code_Complaints.csv', skiprows = [1]

vacant = pd.read_csv('Vacant_and_Abandoned_Buildings_Reported.csv',
        names = ['Type of Service Request', 'Service Request Number', 'Cr
       'Location of Building on the lot', 'Is the Building Dangerous or H
       'Is Building Open Or Board?', 'IF THE BUILDING IS OPEN, WHERE IS T
       'IS THE BUILDING CURRENTLY VACANT OR OCCUPIED?', 'IS THE BUILDING
       'ANY PEOPLE USING PROPERTY? (HOMELESS, CHILDEN, GANGS)',
       'ADDRESS STREET NUMBER', 'ADDRESS STREET DIRECTION',
       'ADDRESS STREET NAME', 'ADDRESS STREET SUFFIX', 'ZIP Code',
       'X Coordinate', 'Y Coordinate', 'Ward', 'Police District',
       'Community Area', 'Latitude', 'Longitude', 'Location'], skiprows =
```

2 Generate summary statistics for these requests including but not limited to number of requests

2.Generate summary statistics for these requests including but not limited to number of requests of each type (and subtype within each of the types above) over time, by neighborhood, response time by the city. Please use a combination of tables and graphs to present these summary stats.

In [7]:
```
#Help function (1)

#### figure and table of total number of request by the four different 31
def main_type_by_year():
    '''
    Input a datafram(string format) and get the summary statistics and th
    311 request from year 2011 to 2016

    Input:
    'graffiti' or 'pot' or 'sanitation'

    Output:
    summary statistics of certain types 311 request by years(2011-2016) u

    '''

    graffiti['year'] = graffiti['Creation Date'].apply(lambda x: x[-4:])
    grouped_graffiti = (graffiti.groupby('year').size())
    #only get the amount from 2011 to 2016
    amount_graffiti = grouped_graffiti[-7:-1]


    pot['year'] = pot['Creation Date'].apply(lambda x: x[-4:])
    grouped_pot = pot.groupby('year').size()
    #only get the amount from 2011 to 2016
    amount_pot = grouped_pot[-7:-1]

    sanitation['year'] = sanitation['Creation Date'].apply(lambda x: x[-4
    grouped_sanitation = sanitation.groupby('year').size()
    #only get the amount from 2011 to 2016
    amount_sanitation = grouped_sanitation[-7:-1]


    vacant['year'] = vacant['Creation Date'].apply(lambda x: x[-4:])
    grouped_vacant = vacant.groupby('year').size()
    #only get the amount from 2011 to 2016
    amount_vacant = grouped_vacant[-7:-1]

    df =  pd.concat([amount_graffiti, amount_pot, amount_sanitation, amou
    df = df.rename(index=str, columns={0: "amount_graffiti", 1: "amount_p

    ##plot
    fig, ax = plt.subplots(figsize =(8,8))
    plt.plot(df.index, df.amount_graffiti, marker = 'D',label = 'amount o
    plt.plot(df.index, df.amount_pot, marker = 'o',label = 'amount of pot
    plt.plot(df.index, df.amount_sanitation, marker ='o', color = 'b',lab
```

```
    plt.plot(df.index, df.amount_vacant, marker = 'D', color = 'maroon',l

    plt.title('311 requests amounts from 2011 to 2016', fontsize=20)
    plt.xlabel(r'year')
    plt.ylabel(r'amount of 311 request')
    plt.legend(loc='upper right')

    plt.show()
    plt.close()

    return df
```

In [8]:
```
#Help function(2-3)


#number of requests of subtypes function
def subtype_request(df):
    '''
    Input a datafram(string format) and get the summary statistics and th
    under this main type.

    Input:
    'graffiti' or 'pot' or 'sanitation'

    Output:

    summary statistics of each types under the input datafram
    '''
    total = len(df)
    if df == 'graffiti':
        grouped_1 = graffiti.groupby('What Type of Surface is the Graffit
        grouped_2 = graffiti.groupby('Where is the Graffiti located?').si
        grouped_3 = graffiti.groupby('Community Area').size().order(ascen
        grouped_4 = graffiti.groupby('response time(days)').size().order(

    elif df == 'pot':
        grouped_1 = pot.groupby('Current Activity').size().order(ascendin
        grouped_2 = pot.groupby('Number of Potholes Filled On Block').siz
        grouped_3 = pot.groupby('Community Area').size().order(ascending=
        grouped_4 = pot.groupby('response time(days)').size().order(ascen
    elif df == 'sanitation':
        grouped_1 = sanitation.groupby('What is the Nature of this Code V
        grouped_2 = sanitation.groupby('Status').size().order(ascending=F
        grouped_3 = sanitation.groupby('Community Area').size().order(asc
        grouped_4 = sanitation.groupby('response time(days)').size().orde

    fig = plt.figure(figsize = (10,10))

    ax1 = fig.add_subplot(2, 2, 1)
```

```
    ax1.title.set_text('{}'.format(grouped_1.index.name))
    grouped_1.plot(kind = 'bar')

    ax2 = fig.add_subplot(2, 2, 2)
    ax2.title.set_text('{}'.format(grouped_2.index.name))
    grouped_2.plot(kind = 'bar')

    ax3 = fig.add_subplot(2, 2, 3)
    ax3.title.set_text('{}'.format(grouped_3.index.name))
    grouped_3.plot(kind = 'bar')

    ax4 = fig.add_subplot(2, 2, 4)
    ax4.title.set_text('{}'.format(grouped_4.index.name))
    grouped_4.plot(kind = 'bar')
    plt.show()
    plt.close()

    return grouped_1, grouped_2, grouped_3, grouped_4



##response time function
def response_time(df):
    '''
    Input a dataframe and add columns of response time(days) depends on c
    creation date. If the one of the date of creation or completion is na
    will be default -1

    Input:
    dataframe
    '''
    date_format = "%m/%d/%Y"

    lst_completion = []
    for i in df['Completion Date']:
        if isinstance(i, str):
            dt = datetime.strptime(i, date_format)
        else:
            dt = np.nan
        lst_completion.append(dt)

    lst_creation = []
    for i in df['Creation Date']:
        if isinstance(i, str):
            dt = datetime.strptime(i, date_format)
        else:
            dt = np.nan
        lst_creation.append(dt)

    days = np.zeros(len(lst_completion))
    for i in range(len(lst_completion)):
```

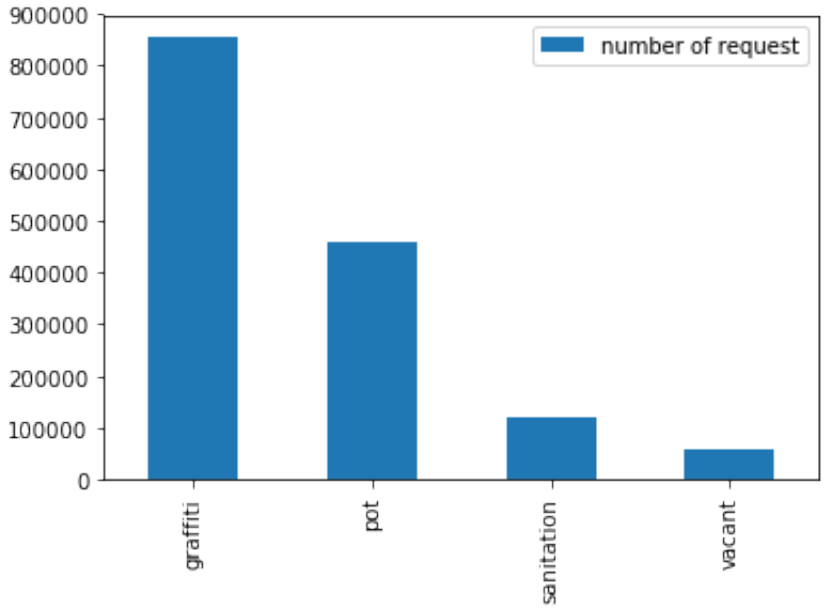
```
    for i in range(len(lst_completion)):
        if isinstance(lst_completion[i], float) or isinstance(lst_creatio
            days[i] = -1
        else:
            days[i] = (lst_completion[i] - lst_creation[i]).days

    df['response time(days)'] = days
    return
```

In [9]:
```
##update the three dataframe with response time
response_time(graffiti)
response_time(pot)
response_time(sanitation)
```

In [10]:
```
## figure and table of total number of request by the four different 311
fig = plt.figure()
dic_total_request = {'graffiti':len(graffiti), 'pot':len(pot), 'sanitatio
df_total_request = pd.Series(dic_total_request).to_frame('number of reque
df_total_request.plot(kind = 'bar')
plt.show()
```

```
<matplotlib.figure.Figure at 0x11bbb9f60>
```



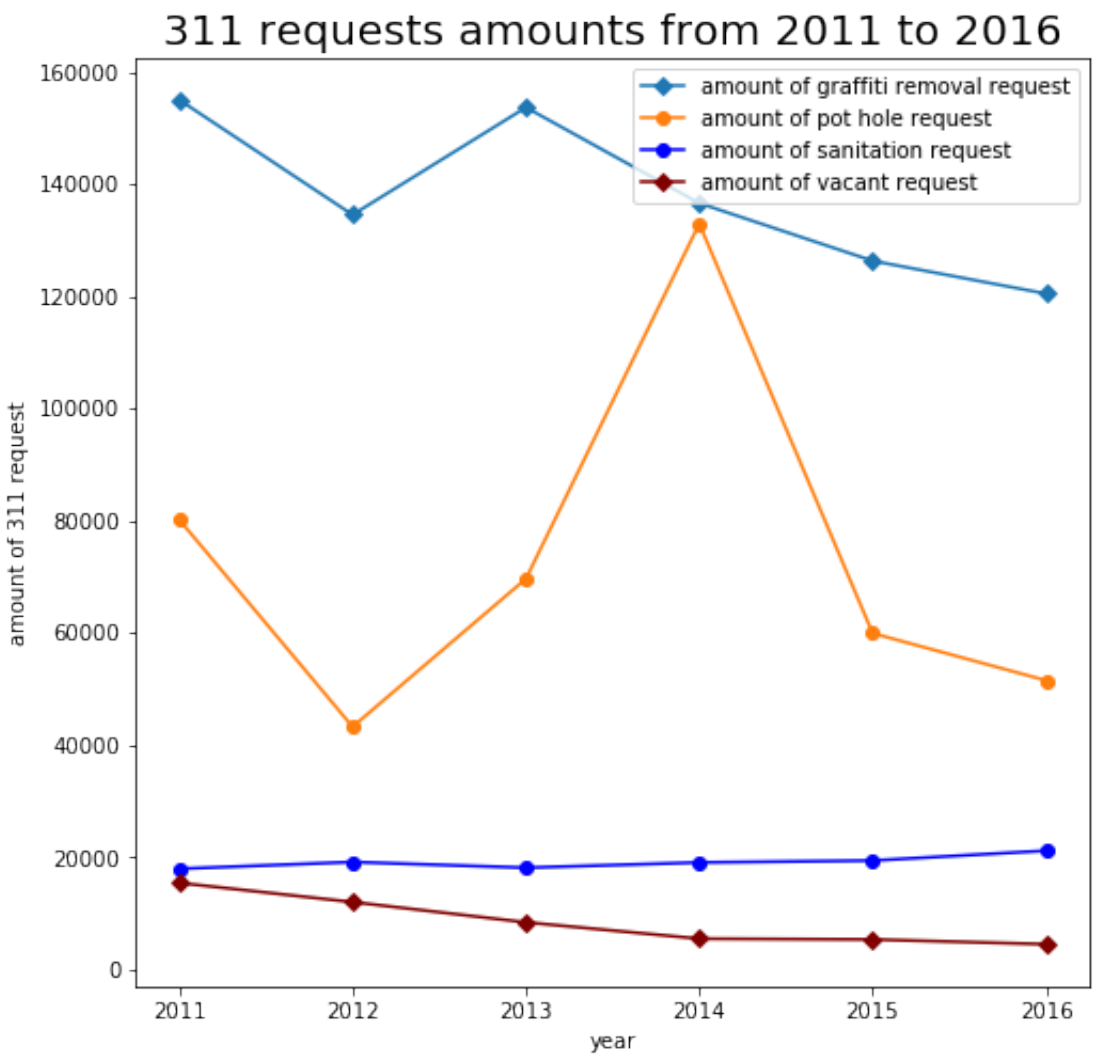
In [11]:
```
#table form
df_total_request
```

Out[11]:

	number of request
graffiti	855697
pot	458469
sanitation	118923
vacant	58745

From the table and bar chart plot of the four types of 311 request, we could know that the graffiti requests' number is the largest over time, which is equal to 855697. The vacant requests' number is the smallest, which equal to 58745.However, this result come from the data which has not groupby a constant time, so it's not that representative. So next, we will only focus on the 2011- 2016 311 requests of this four types.

In [14]:
```
df = main_type_by_year()
```



In [15]:
```
df
```

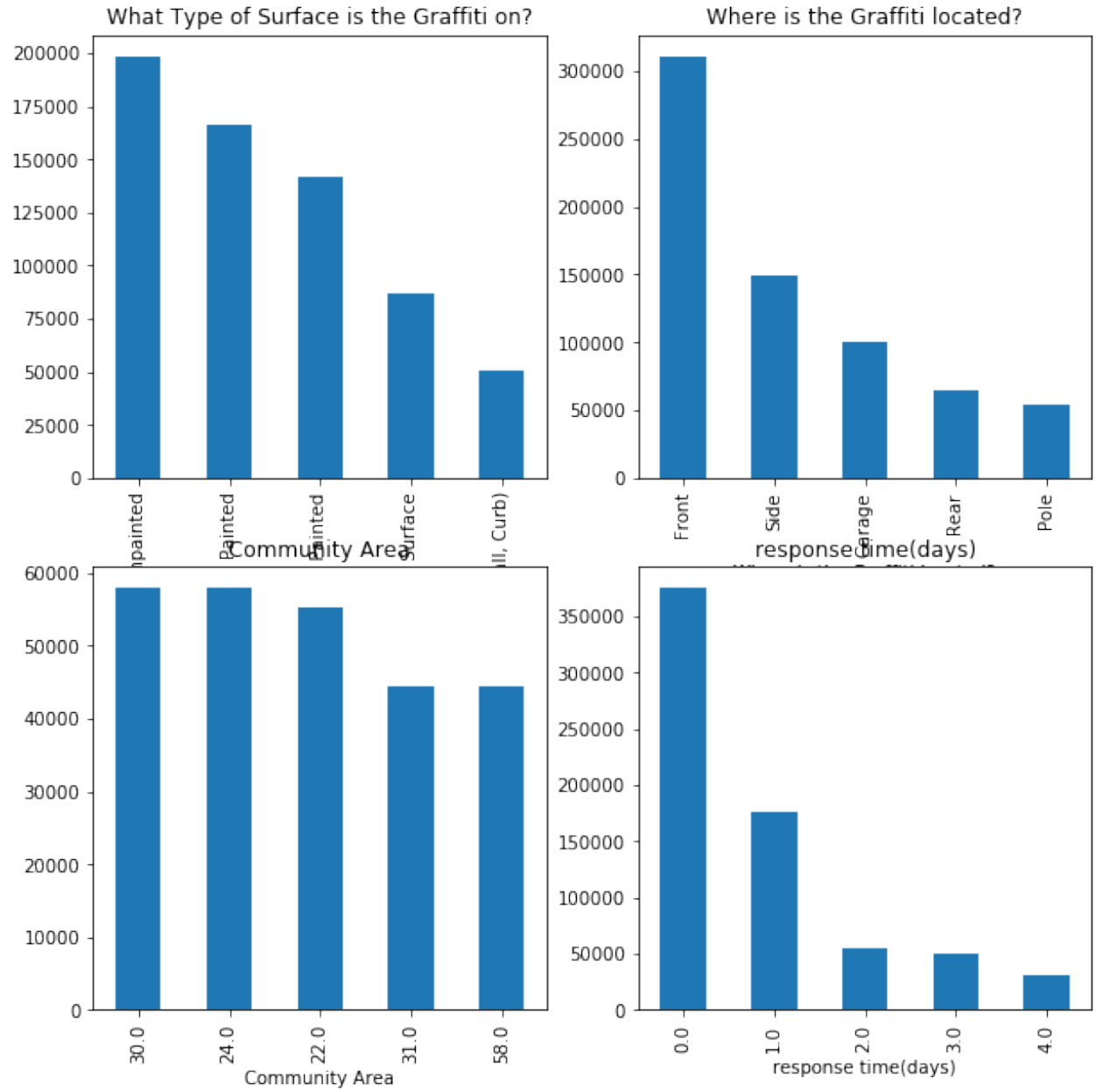
Out[15]:

	amount_graffiti	amount_pot	amount_sanitation	amount_vacant
year				
2011	155019	80162	17922	15395
2012	134529	43237	19124	11984
2013	153669	69630	18112	8370
2014	136595	132938	19042	5451
2015	126310	59910	19382	5313
2016	120461	51507	21139	4472

From the table we could notice that the 311 requests of different year show different pattern. Take a look at the line plot above, graffiti removel requests and vacant request are roughly decrease from 2011 to 2016. After group by year, the yearly number of graffiti removel 311 requests is also the largest one, number of vacant request is still the smallest. Amount of sanitation request seems not change a lot in different year. However, pot hole request increased sharply in 2014, although it still smaller than graffiti removel request.

In [16]:
```
##number of request in each subtypes including response time
#graffiti subtype
graffiti_summary = subtype_request('graffiti')
```

```
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:19: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:20: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:21: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:22: FutureWarning: order is deprecated, use sort_values(...)
```



In [17]:
```
graffiti_summary[0].to_frame('top 5 amount')
```

Out[17]:

	top 5 amount
What Type of Surface is the Graffiti on?	
Brick - Unpainted	198685
Brick - Painted	166048
Metal - Painted	142147
Other / Unknown Surface	87059
Cement (Sidewalk, Alley, Wall, Curb)	50464

In [18]:
```
graffiti_summary[1].to_frame('top 5 amount')
```

Out[18]:

	top 5 amount
Where is the Graffiti located?	
Front	310846
Side	148707
Garage	100115
Rear	64015
Pole	54621

In [19]:
```
graffiti_summary[2].to_frame('top 5 amount')
```

Out[19]:

	top 5 amount
Community Area	
30.0	57906
24.0	57892
22.0	55115
31.0	44366
58.0	44281

In [20]:
```
graffiti_summary[3].to_frame('top 5 amount')
```

Out[20]:

	top 5 amount
response time(days)	
0.0	375372
1.0	176554
2.0	55248
3.0	50379
4.0	31064

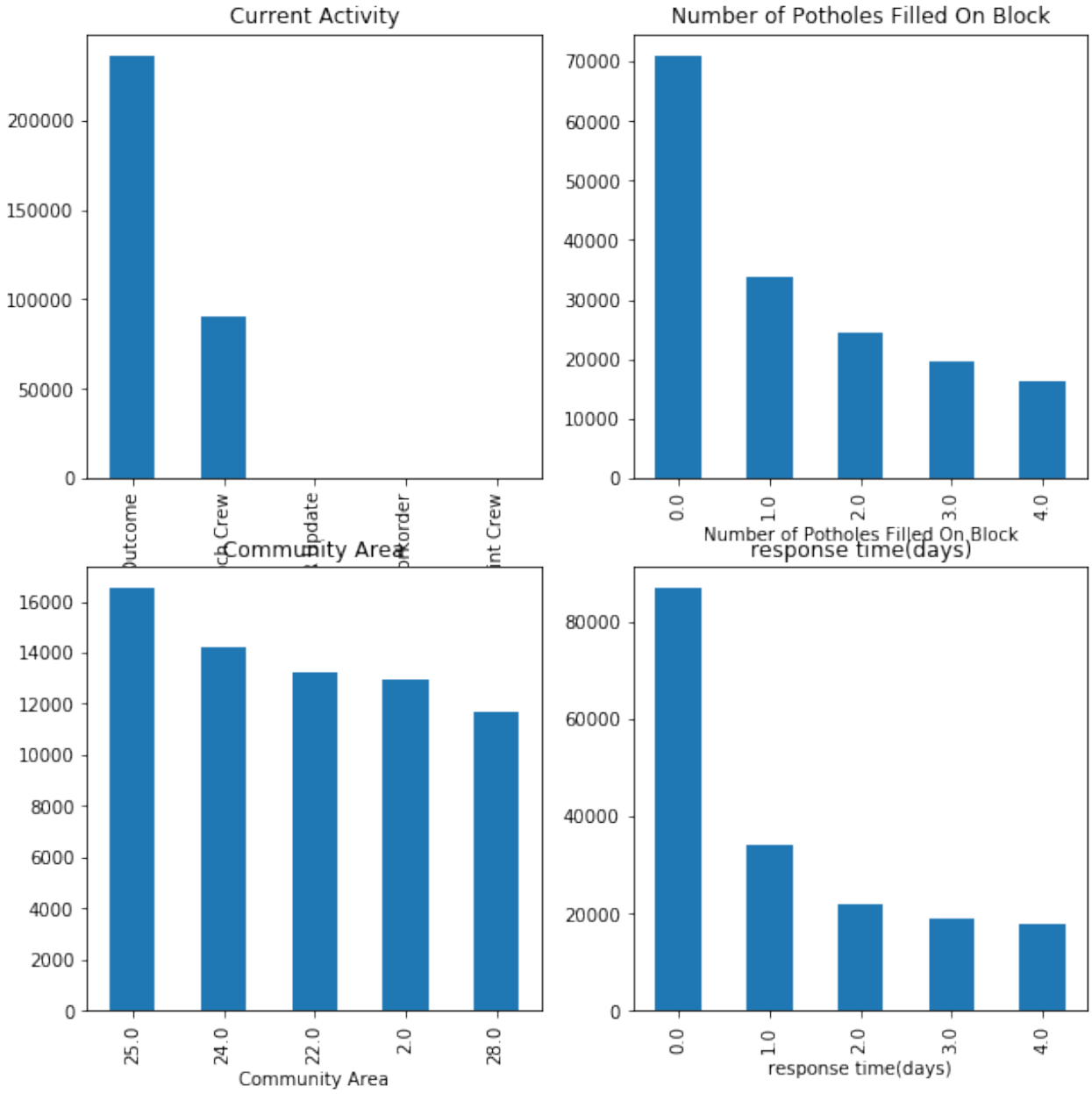
From the plot and table, we could know that the top 5 amount of type of surface is the graffiti on from 311 graffiti removal data is as above. Most request about graffiti removel is brick-unpainted and brick painted, which means brick is the most usual graffiti area. Most graffiti located in the front and it's amount much larger than other kind of location. This mean front brick is always unprotectable.

In terms of community area, the graffiti removal requests do not have much difference. 30 community area has the most graffiti, 57906.

Take a look at the response time(days), most graffiti removel requests will complete the task the day that it has been request.

In [21]:
```
sub_pot = subtype_request('pot')
```

```
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:25: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:26: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:27: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:28: FutureWarning: order is deprecated, use sort_values(...)
```



In [22]:
```
sub_pot[0].to_frame('top 5 amount')
```

Out[22]:

	top 5 amount
Current Activity	
Final Outcome	236378
Dispatch Crew	90117
Follow-on SR Update	324
Generate Workorder	188
Complaint Crew	26

In [23]:
```
sub_pot[1].to_frame('top 5 amount')
```

Out[23]:

	top 5 amount
Number of Potholes Filled On Block	
0.0	71044
1.0	33665
2.0	24415
3.0	19601
4.0	16343

In [24]:
```
sub_pot[2].to_frame('top 5 amount')
```

Out[24]:

	top 5 amount
Community Area	
25.0	16521
24.0	14232
22.0	13228
2.0	12965
28.0	11713

In [25]:
```
sub_pot[3].to_frame('top 5 amount')
```

Out[25]:

	top 5 amount
response time(days)	
0.0	86909
1.0	34118
2.0	22017
3.0	19031
4.0	17908

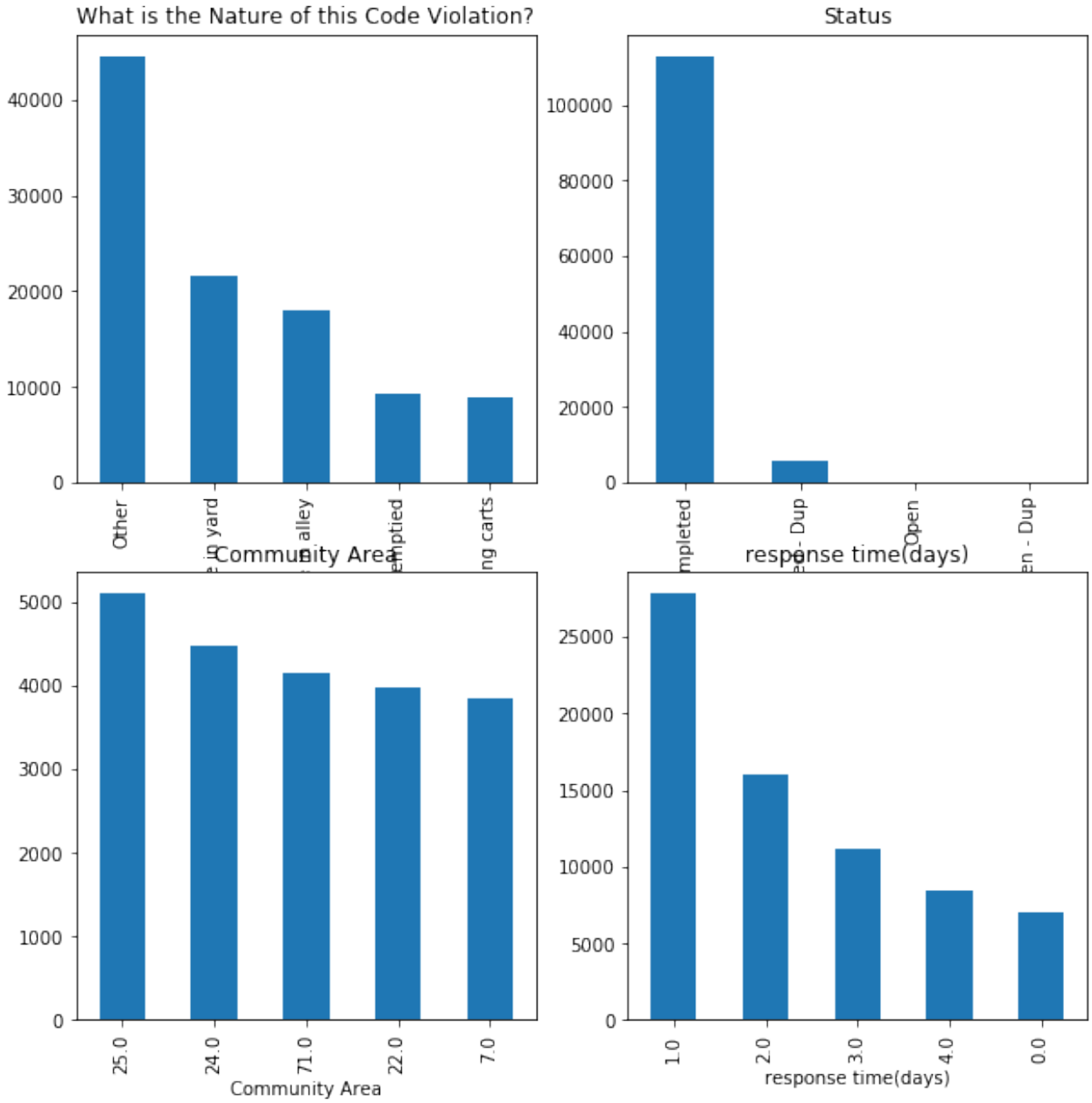
From the plot and table, we could know that the top 5 amount of current activity from 311 potholes data is as above. Most request current activity of potholes request is Final Outcome and Dispatch Crew. Most pot holes don't have hole filled on block.

In terms of community area, the pot holes requests have slightly difference compared with graffiti requests. 25 community area has the most pot hole requests, 16521. The 24 community area has 14232 pot hole requests and 57892 graffiti requests, rank 2nd in both graffiti removel request and pot holes request.

Take a look at the response time(days), most pot holes requests will complete the task the day that it has been request.

In [26]:
```
sub_sanitation = subtype_request('sanitation')
```

```
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:30: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:31: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:32: FutureWarning: order is deprecated, use sort_values(...)
/Users/zhuoleng/anaconda/lib/python3.5/site-packages/ipykernel/__main_
_.py:33: FutureWarning: order is deprecated, use sort_values(...)
```



In [27]:
```
sub_sanitation[0].to_frame('top 5 amount')
```

Out[27]:

	top 5 amount
What is the Nature of this Code Violation?	
Other	44595
Garbage in yard	21556
Garbage in alley	18034
Dumpster not being emptied	9221
Overflowing carts	8942

In [28]:
```
sub_sanitation[1].to_frame('top 5 amount')
```

Out[28]:

	top 5 amount
Status	
Completed	112974
Completed - Dup	5658
Open	172
Open - Dup	119

In [29]:
```
sub_sanitation[2].to_frame('top 5 amount')
```

Out[29]:

	top 5 amount
Community Area	
25.0	5103
24.0	4476
71.0	4158
22.0	3971
7.0	3848

In [30]:
```
sub_sanitation[3].to_frame('top 5 amount')
```

Out[30]:

	top 5 amount
response time(days)	
1.0	27841
2.0	15990
3.0	11221
4.0	8491
0.0	7086

From the plot and table, we could know that the top 5 amount of types of nature violation from 311 sanitation data is as above. Most request can not get a accurate decision of the nature of violation, so their category is under OTHER.

In terms of community area, the 25, 24 22 community areas are in the top 5 amount list again. These community areas are also in the pot holes requests top 5 community acres request list. community areas 7 and 71 are the first time show here.

Take a look at the response time(days), most sanitation requests will complete the task the day after it has been request. Compare to the graffiti removel request and pot hole request, sanitation request takes longer time to response.

Problem 2: Data Augmentation and APIs

In [37]:
```
##help function to get census api

def get_census(dataframe):
    '''
    Use lat and lon of dataframe to get fips number and use fips number t
    census api

    Input: dataframe

    Output: dataframe
    '''

    df = dataframe[(dataframe['year'] == '2017')].reset_index(drop = True
    df = df.dropna(subset = [['Latitude','Longitude']]).reset_index(drop

    lat = df['Latitude']
    lon = df['Longitude']
```

```
    B01001F_001E = []
    B19119_001E = []
    B05010_001E = []

    for i in range(len(lat)):

        #first use lat and lon to get fips data
        api_url = 'http://data.fcc.gov/api/block/find?format=json&latitud
        r = requests.get(api_url)
        soup = bs4.BeautifulSoup(r.text, 'lxml')
        fips = json.loads(soup.text)

        state = fips['State']['FIPS']
        county = fips['County']['FIPS'][2:]
        tract = fips['Block']['FIPS'][5:-4]
        block = fips['Block']['FIPS'][-4:]

        #then use fips data to call census api
        new_url = 'http://api.census.gov/data/2014/acs5?get=B01001F_001E,

        r = requests.get(new_url)
        soup = bs4.BeautifulSoup(r.text, 'lxml')
        census = json.loads(soup.text)[1]

        B01001F_001E.append(census[0])
        B19119_001E.append(census[1])
        B05010_001E.append(census[2])

    df['B01001F_001E'] = B01001F_001E
    df['B19119_001E'] = B19119_001E
    df['B05010_001E'] = B05010_001E

    return df
```

In [39]:
```
import json
import sys
import csv
import json
import time
import requests
import bs4
```

In [50]:
```
def get_census(dataframe):
    '''
    Use lat and lon of dataframe to get fips number and use fips number t
```

```
    use lat and lon of dataframe to get fips number and use fips number t
    census api

    Input: dataframe

    Output: dataframe
    '''
    df = dataframe[(dataframe['year'] == '2017')].reset_index(drop = True
    df = df.dropna(subset = [['Latitude','Longitude']]).reset_index(drop

    lat = df['Latitude']
    lon = df['Longitude']

    B01001F_001E = []
    B19119_001E = []
    B05010_001E = []

    for i in range(len(lat)):

        #first use lat and lon to get fips data
        api_url = 'http://data.fcc.gov/api/block/find?format=json&latitud
        r = requests.get(api_url)
        soup = bs4.BeautifulSoup(r.text, 'lxml')
        fips = json.loads(soup.text)

        state = fips['State']['FIPS']
        county = fips['County']['FIPS'][2:]
        tract = fips['Block']['FIPS'][5:-4]
        block = fips['Block']['FIPS'][-4:]

        #then use fips data to call census api
        new_url = 'http://api.census.gov/data/2014/acs5?get=B01001F_001E,

        r = requests.get(new_url)
        soup = bs4.BeautifulSoup(r.text, 'lxml')
        census = json.loads(soup.text)[1]

        B01001F_001E.append(census[0])
        B19119_001E.append(census[1])
        B05010_001E.append(census[2])

    df['B01001F_001E'] = B01001F_001E
    df['B19119_001E'] = B19119_001E
    df['B05010_001E'] = B05010_001E

    return df
```

In [52]:
```
census_sanitation = get_census(sanitation)
census_vacant = get_census(vacant)
```

In [51]:
```
census_sanitation
```

Out[51]:

	Creation Date	Status	Completion Date	Service Request Number	Type of Service Request	What is the Nature of this Code Violation?	Street Addres
0	01/01/2017	Completed	01/02/2017	17-00004522	Sanitation Code Violation	Garbage in alley	5203 N RIVERS TER
1	01/03/2017	Completed - Dup	01/03/2017	17-00044149	Sanitation Code Violation	Garbage in alley	1310 N SALLE
2	01/03/2017	Completed - Dup	01/03/2017	17-00044193	Sanitation Code Violation	Garbage in yard	1310 N SALLE
3	01/03/2017	Completed - Dup	01/03/2017	17-00050708	Sanitation Code Violation	Dumpster not being emptied	4400 S DREXE BLVD
4	01/03/2017	Completed	01/03/2017	17-00039892	Sanitation Code Violation	Dumpster not being emptied	74 W HARRIS ST
5	01/03/2017	Completed	01/03/2017	17-00050710	Sanitation Code Violation	NaN	4733 W FOSTE
6	01/03/2017	Completed	01/03/2017	17-00051370	Sanitation Code Violation	Garbage in alley	1735 W DIVISIC
7	01/03/2017	Completed	01/03/2017	17-00051528	Sanitation Code Violation	Garbage in alley	3129 W AUGUS BLVD
8	01/03/2017	Completed	01/03/2017	17-00052442	Sanitation Code Violation	Standing water	3000 N HOYNE
9	01/01/2017	Completed - Dup	01/03/2017	17-00011372	Sanitation Code Violation	Overflowing carts	4048 N MOZAR

10	01/01/2017	Completed - Dup	01/03/2017	17-00011373	Sanitation Code Violation	Overflowing carts	4048 N MOZAR
11	01/01/2017	Completed - Dup	01/03/2017	17-00011532	Sanitation Code Violation	Overflowing carts	4032 N MOZAR
12	01/01/2017	Completed - Dup	01/03/2017	17-00011533	Sanitation Code Violation	Other	4048 N MOZAR
13	01/01/2017	Completed	01/03/2017	17-00003196	Sanitation Code Violation	Dumpster not being emptied	5500 W ST
14	01/01/2017	Completed	01/03/2017	17-00003217	Sanitation Code Violation	Dumpster not being emptied	5508 W ST
15	01/01/2017	Completed	01/03/2017	17-00007175	Sanitation Code Violation	Garbage in yard	6450 N DAMEN
16	01/01/2017	Completed	01/03/2017	17-00009824	Sanitation Code Violation	Other	7689 W FORES PRESE AVE
17	01/01/2017	Completed	01/03/2017	17-00011529	Sanitation Code Violation	Overflowing carts	4028 N MOZAR
18	01/01/2017	Completed	01/03/2017	17-00011531	Sanitation Code Violation	Overflowing carts	4032 N MOZAR
19	01/01/2017	Completed	01/03/2017	17-00012079	Sanitation Code Violation	Dumpster not being emptied	2805 W LAWRE AVE
20	01/01/2017	Completed	01/03/2017	17-00012111	Sanitation Code Violation	Garbage in alley	4765 N VIRGIN
21	01/02/2017	Completed	01/03/2017	17-00012728	Sanitation Code Violation	Garbage in alley	6156 S KARLO

22	01/02/2017	Completed	01/03/2017	17-00015443	Sanitation Code Violation	Overflowing carts	2433 N ASHLA AVE
23	01/02/2017	Completed	01/03/2017	17-00017008	Sanitation Code Violation	Dumpster not being emptied	4400 S DREXE BLVD
24	01/02/2017	Completed	01/03/2017	17-00019975	Sanitation Code Violation	Other	445 N RIDGE AVE
25	01/02/2017	Completed	01/03/2017	17-00023515	Sanitation Code Violation	Garbage in yard	4240 N KEELE
26	01/02/2017	Completed	01/03/2017	17-00025284	Sanitation Code Violation	Garbage in alley	1310 N SALLE
27	01/02/2017	Completed	01/03/2017	17-00025623	Sanitation Code Violation	Garbage in alley	1043 E ST
28	01/02/2017	Completed	01/03/2017	17-00026533	Sanitation Code Violation	Garbage in yard	4140 N PITTSE AVE
29	01/02/2017	Completed	01/03/2017	17-00028451	Sanitation Code Violation	Garbage in yard	1501 W SCHOO
...	...	...	...	...	...	...	...
3987	03/27/2017	Open	NaN	17-01710207	Sanitation Code Violation	Garbage in yard	2425 W LEXING ST
3988	03/27/2017	Open	NaN	17-01710345	Sanitation Code Violation	Construction Site Cleanliness/Fence	1351 S KARLO
3989	03/27/2017	Open	NaN	17-01711150	Sanitation Code Violation	Garbage in yard	2432 W LEXING ST
3990	03/27/2017	Open	NaN	17-01712845	Sanitation Code Violation	Garbage in yard	6920 S ASHLA AVE

3991	03/27/2017	Open	NaN	17-01713120	Sanitation Code Violation	Overflowing carts	8000 S WASHT AVE
3992	03/27/2017	Open	NaN	17-01713285	Sanitation Code Violation	Other	5234 S HONOI
3993	03/27/2017	Open	NaN	17-01713395	Sanitation Code Violation	Other	8555 S CONST AVE
3994	03/27/2017	Open	NaN	17-01713550	Sanitation Code Violation	Other	3552 S DEARE ST
3995	03/27/2017	Open	NaN	17-01714250	Sanitation Code Violation	Dumpster not being emptied	3138 W CERMA
3996	03/27/2017	Open	NaN	17-01715049	Sanitation Code Violation	Garbage in yard	7531 S NORM/
3997	03/27/2017	Open	NaN	17-01715239	Sanitation Code Violation	Dog feces in yard	7712 S RIDGEI AVE
3998	03/27/2017	Open	NaN	17-01715425	Sanitation Code Violation	Other	5949 S PEORI/
3999	03/27/2017	Open	NaN	17-01720569	Sanitation Code Violation	Other	2330 W 112TH
4000	03/27/2017	Open	NaN	17-01720790	Sanitation Code Violation	Garbage in yard	928 W ST
4001	01/14/2017	Open - Dup	NaN	17-00254971	Sanitation Code Violation	Other	4028 N MOZAI
4002	01/16/2017	Open - Dup	NaN	17-00288340	Sanitation Code Violation	Overflowing carts	4028 N MOZAI
		Open -		17-	Sanitation		4048 N

4003	01/16/2017	Dup	NaN	00288342	Code Violation	Overflowing carts	MOZAR
4004	01/18/2017	Open - Dup	NaN	17-00311993	Sanitation Code Violation	Garbage in yard	1439 S LAWNDI AVE
4005	02/09/2017	Open - Dup	NaN	17-00721030	Sanitation Code Violation	Overflowing carts	448 W JAMES
4006	03/08/2017	Open	NaN	17-01289908	Sanitation Code Violation	Other	3038 S WALLA
4007	03/13/2017	Open	NaN	17-01385167	Sanitation Code Violation	Garbage in yard	6935 S CAMPE AVE
4008	03/14/2017	Open	NaN	17-01395498	Sanitation Code Violation	Garbage in yard	4929 W WEST I AVE
4009	03/16/2017	Open	NaN	17-01430498	Sanitation Code Violation	Garbage in yard	3038 S WALLA
4010	03/16/2017	Open	NaN	17-01431396	Sanitation Code Violation	Overflowing carts	3214 S UNION
4011	03/16/2017	Open	NaN	17-01436774	Sanitation Code Violation	Other	6347 S WOLC AVE
4012	03/17/2017	Completed - Dup	03/31/2017	17-01453016	Sanitation Code Violation	Other	700 N MICHIC AVE
4013	03/17/2017	Open - Dup	NaN	17-01447340	Sanitation Code Violation	Construction Site Cleanliness/Fence	846 W WEBST AVE
4014	03/17/2017	Open	NaN	17-01448708	Sanitation Code Violation	Garbage in alley	2803 S ARCHE
4015	03/17/2017	Open	NaN	17-	Sanitation Code	Construction Site	2816 S

				01459360	Violation	Cleanliness/Fence	KEELE
4016	03/18/2017	Open - Dup	NaN	17-01474064	Sanitation Code Violation	Construction Site Cleanliness/Fence	846 W WEBST AVE

4017 rows × 21 columns

In [53]:
```
census_vacant
```

Out[53]:

	Type of Service Request	Service Request Number	Creation Date	Location of Building on the lot	Is the Building Dangerous or Hazarous?	Is Building Open Or Board?	IF THE BUILDING I OPEN, WHERE IS THE ENTRY POINT?
0	Vacant/Abandoned Building	17-00042635	01/03/2017	Front	NaN	Open	NaN
1	Vacant/Abandoned Building	17-00068420	01/04/2017	Front	NaN	Open	NaN
2	Vacant/Abandoned Building	17-00068968	01/04/2017	Front	NaN	Open	NaN
3	Vacant/Abandoned Building	17-00070421	01/04/2017	Front	NaN	Open	NaN

1. What types of blocks get “Sanitation Code Complaints”?

#B01001F_001E: RACE!!Total population

#B19119_001E : INCOME AND BENEFITS (IN 2014 INFLATION-ADJUSTED DOLLARS)!!Total households!!Median household income (dollars)

#B05010_001E: HOUSEHOLDS BY TYPE!!Total households!!Average family size

In [54]:
```
census_sanitation['B01001F_001E'].astype(float).describe()
```

Out[54]:
```
count    4017.000000
mean      481.724421
std       803.206220
min         0.000000
25%        12.000000
50%       121.000000
75%       518.000000
max      4937.000000
Name: B01001F_001E, dtype: float64
```

In [55]:
```
census_sanitation['B19119_001E'].astype(float).describe()
```

Out[55]:
```
count      4017.000000
mean      62764.394324
std       45731.757192
min        7019.000000
25%       33603.000000
50%       45625.000000
75%       72500.000000
max      250001.000000
Name: B19119_001E, dtype: float64
```

In [56]:
```
census_sanitation['B05010_001E'].astype(float).describe()
```

Out[56]:
```
count    4017.000000
mean      874.692059
std       530.494858
min        22.000000
25%       462.000000
50%       756.000000
75%      1193.000000
max      2513.000000
Name: B05010_001E, dtype: float64
```

so from the describe data table, we could know that the the total population in Sanitation Code Complaints block are between(0, 4937). Most populaion of block is around 121(median), which is much larger than that in vacant reported block. Also, from the mean number of population = 481, we could testify that the vacant reported block has less population than sanitation complaints block.

Then take a look at Median Family Income in the past 12 months. The median income is between (7019, 250001). The mean is 62764, compared to that of vacant (39391), the block has higher median family income in the past 12 month. However, the gap between rich and poor is much

really larger around sanitation reported blocks. The std is almost two times of that of vacant report blocks.

Look at the description data of ratio of income to poverty level in the past 12 months of Children by Living Arrangements and Nativity of Parents. It's means own children under 18 years living in families or subfamilies for whom poverty status Is determined. Frome description data, we could know that the ratio is between (22, 2513), with mean = 874 and median = 756. This data does not seems much different between the two kinds of blocks.

2.What types of blocks get “Vacant and Abandoned Buildings Reported”?

In [57]:
```
census_vacant['B01001F_001E'].astype(float).describe()
```

Out[57]:
```
count    1091.000000
mean      231.932172
std       585.782429
min         0.000000
25%         0.000000
50%        14.000000
75%       122.000000
max      4937.000000
Name: B01001F_001E, dtype: float64
```

In [58]:
```
census_vacant['B19119_001E'].astype(float).describe()
```

Out[58]:
```
count      1091.000000
mean      39391.623281
std       21063.461079
min        7019.000000
25%       26375.000000
50%       34875.000000
75%       46053.000000
max      183889.000000
Name: B19119_001E, dtype: float64
```

In [59]:
```
census_vacant['B05010_001E'].astype(float).describe()
```

Out[59]:
```
count    1091.000000
mean      838.350137
std       458.221576
min        37.000000
25%       514.000000
50%       717.000000
75%      1065.000000
max      2513.000000
Name: B05010_001E, dtype: float64
```

So from the describe data table, we could know that the the total population in vacant block are between(0, 4937). Most populaion of block is around 14(median). So we could know the std is very large in vacant area in terms of population

Then take a look at Median Family Income in the past 12 months. The median income is between (7019, 183889). The mean is 39391 and median is 34875. The gap between the rich and the poor is really huge around vacant block.

Look at the description data of ratio of income to poverty level in the past 12 months of Children by Living Arrangements and Nativity of Parents. It's means own children under 18 years living in families or subfamilies for whom poverty status Is determined. From description data, we could know that the ratio is between (37, 2513), with mean = 838 and median = 717. We could deduct that in vacant block the rich people are more than poor people.

3.Does that change over time in the data you collected?

In [62]:
```
census_sanitation['B01001F_001E'] = census_sanitation['B01001F_001E'].ast
census_sanitation['B19119_001E'] = census_sanitation['B19119_001E'].astyp
census_sanitation['B05010_001E'] = census_sanitation['B05010_001E'].astyp

census_vacant['B01001F_001E'] = census_vacant['B01001F_001E'].astype(floa
census_vacant['B19119_001E'] = census_vacant['B19119_001E'].astype(float)
census_vacant['B05010_001E'] = census_vacant['B05010_001E'].astype(float)
```

In [63]:
```
#get the month data of each type of request
census_sanitation['month'] = census_sanitation['Creation Date'].apply(lam
census_vacant['month'] = census_vacant['Creation Date'].apply(lambda x: x
```

In [65]:
```
#groupby time
census_sanitation.groupby('month').mean()
```

Out[65]:

	ZIP Code	X Coordinate	Y Coordinate	Ward	Police District	Community Area	La
month							
01	60629.344007	1.162720e+06	1.889744e+06	23.798361	12.576230	36.045082	41
02	60629.790205	1.161818e+06	1.891876e+06	24.785819	12.932018	34.319444	41
03	60630.666900	1.161918e+06	1.891360e+06	24.709587	12.589923	35.037089	41

In [66]:
```
census_sanitation.groupby('month').median()
```

Out[66]:

	ZIP Code	X Coordinate	Y Coordinate	Ward	Police District	Community Area	Latitude	Lon
month								
01	60626.0	1.162616e+06	1.896502e+06	25.0	12.0	29.0	41.871850	-87.
02	60628.0	1.160967e+06	1.899278e+06	26.0	12.0	28.0	41.879479	-87.
03	60628.0	1.160744e+06	1.898529e+06	26.0	11.0	28.0	41.877504	-87.

I only use the recent three months' data. After I groupby month(01,02,03), I could know that although the B01001F index seems decrease after Jan.The median of population increase from Jan. In addition, the B19119_001E seems stable during Jan. and Feb.However, it drops from Mar., which means that the median family income decrease since Mar..At last, we look at B05010_001E, it does not has any pattern over time in the data I collected.

In [67]:
```
census_vacant.groupby('month').mean()
```

Out[67]:

	Is the Building Dangerous or Hazarous?	ADDRESS STREET NUMBER	ZIP Code	X Coordinate	Y Coordinate	Ward
month						
01	NaN	5675.690678	60630.482906	1.172548e+06	1.861674e+06	16.697872
02	NaN	5903.407125	60630.061224	1.170796e+06	1.863434e+06	18.175573
03	NaN	5644.417749	60628.717724	1.169605e+06	1.863641e+06	19.212121

In [68]:
```
census_vacant.groupby('month').median()
```

Out[68]:

	Is the Building Dangerous or Hazarous?	ADDRESS STREET NUMBER	ZIP Code	X Coordinate	Y Coordinate	Ward	Police District	Com Area
month								
01	NaN	6330.5	60624.0	1.173752e+06	1.857797e+06	16.0	7.0	50.0
02	NaN	6351.0	60628.0	1.171285e+06	1.857630e+06	17.0	7.0	51.0
03	NaN	6001.0	60628.0	1.171199e+06	1.858839e+06	17.0	7.0	51.0

The vacant reported data does not has any pattern over time in the data I collected.

4.What is the difference in blocks that get “Vacant and Abandoned Buildings Reported” vs “Sanitation Code Complaints”?

Just as I write in the second question. There are mainly two different in blocks that get “Vacant and Abandoned Buildings Reported” vs “Sanitation Code Complaints”? First, the vacant reported block has less population than sanitation complaints block. Second: the vacant reported blocks are more poor than sanitation complaints block.

Problem 3: Data Augmentation and APIs

1.Of the four types of requests you have data for, which request type is the most likely given the

call came from 7500 S Wolcott Ave? What are the probabilities for each type of request?

In [69]:
```
# question 1
address = '7500 S Wolcott Ave'

#From the address, we could know that the zip code of address is 60620
count_graffiti = len(graffiti[graffiti['ZIP Code'] == 60620.0])
count_pot = len(pot[pot['ZIP Code'] == 60620.0])
count_sanitation = len(sanitation[sanitation['ZIP Code'] == 60620.0])
count_vacant = len(vacant[vacant['ZIP Code'] == 60620.0])

total = count_graffiti + count_pot + count_vacant + count_sanitation
p_graffiti = count_graffiti/total
p_pot = count_pot/total
p_sanitation = count_sanitation/total
p_vacant = count_vacant/total
```

In [71]:
```
p_graffiti
```

Out[71]:
```
0.07493904297470283
```

In [72]:
```
p_pot
```

Out[72]:
```
0.557375800060957
```

In [73]:
```
p_sanitation
```

Out[73]:
```
0.22908412069491008
```

In [74]:
```
p_vacant
```

Out[74]:
```
0.13860103626943004
```

The probability of each type of request is as above.

2.Let’s now assume that a call comes in about Graffiti Removal. Which is more likely – that the call came from Lawndale or Uptown? How much more or less likely is it to be from Lawndale versus Uptown?

In [75]:
```
#Because Lawndale is one of the community area of Chicago and it communit
#area code is 03.
count_graffiti_law = len(graffiti[graffiti['Community Area'] == 30.0])
count_graffiti_up = len(graffiti[graffiti['Community Area'] == 03.0])

total = count_graffiti_up + count_graffiti_law
dif = count_graffiti_law/total - count_graffiti_up/total
```

In [77]:
```
count_graffiti_law
```

Out[77]: 57906

In [78]:
```
count_graffiti_up
```

Out[78]: 11913

In [79]:
```
dif
```

Out[79]: 0.6587461865681261

The call for Graffiti Removal is more likely came from Lawndale. The probability that Lawndale is more likely is 0.6587461865681261

3.Now assume that you don’t have access to all the raw data and you know the following things:

There are a total of 1000 calls, 600 from Englewood and 400 from Uptown. Of the 600 calls from Englewood, 100 of them are about Graffiti Removal. Of the 400 calls from Uptown, 160 are about Graffiti Removal. If a call comes about Graffiti Removal, how much more/less likely is it that the call came from Englewood versus Uptown?

In [80]:
```
p_eng = 100/600
p_uptown = 160/400
dif_2 = p_uptown - p_eng
```

In [81]:
```
dif_2
```

Out[81]: 0.23333333333333336

There is 0.23333333333333336 less likely that the call came from Englewood versus Uptown

In [ ]:
```

```