In [1]:
```
import numpy as np
import seaborn as sns
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
```

In [2]:
```
%matplotlib inline
sns.set(style="darkgrid")
```

Problem 1 - Analyzing 311 Data

In [3]:
```
# Used the following datasets from Chicago Open data portal
files = ["311_Service_Requests_-_Graffiti_Removal.csv",
         "311_Service_Requests_-_Pot_Holes_Reported.csv",
         "311_Service_Requests_-_Sanitation_Code_Complaints.csv",
         "311_Service_Requests_-_Vacant_and_Abandoned_Buildings_Reporte
d.csv"]
hhs_file = "Census_Data_-_Selected_socioeconomic_indicators_in_Chicago__
2008___2012.csv"
```

In [4]:

```
# Read data and massage the data
frames = [ pd.read_csv(file) for file in files ]
for frame in frames:
    frame.columns = map(str.lower, frame.columns)
df = pd.concat(frames)

# combine zip and zip code
df['zip code'].fillna(df['zip'], inplace=True)
del df['zip']

# drop other columns
df.drop(df.columns[[0,1,2,3,4,9,10,11,12,13,14,17]], axis=1, inplace=Tru
e)

# drop rows with NANs in essential columns
df.dropna(subset=["community area","zip code","ward","police district",
"creation date", "completion date", "latitude"], inplace=True)

# combine status
df.replace("Pot Hole in Street","Pothole in Street", inplace=True)

# convert to categories
cat = df[["type of service request", "status", "what type of surface is
 the graffiti on?"]]
df[["type of service request", "status", "what type of surface is the gr
affiti on?"]] = cat.apply(lambda x: x.astype('category'))

# convert to integer
ints = df[["community area", "police district", "ward", "zip code"]]
df[["community area","police district", "ward", "zip code"]] = ints.appl
y(lambda x: x.astype('int'))

# convert to datetime
df['creation date'] = pd.to_datetime(df['creation date'],infer_datetime_
format=True)
df['completion date'] = pd.to_datetime(df['completion date'],infer_datet
ime_format=True)
df['response time'] = ((df['completion date'] - df['creation date']) / n
p.timedelta64(1, 'D')).astype(int)

# combine with Chicago HHS data
hhs = pd.read_csv(hhs_file)
hhs.columns = map(str.lower, hhs.columns)
hhs.dropna(subset=["community area number"], inplace=True)
hhs[["community area number"]] = hhs[["community area number"]].apply(la
mbda x: x.astype('int'))

df = pd.merge(df, hhs, how='inner', left_on="community area",
right_on="community area number")

#extract year
df['year'] = df['creation date'].dt.year

df.columns
```

```
//anaconda/lib/python3.6/site-packages/IPython/core/interactiveshell.p
y:2821: DtypeWarning: Columns (4,10) have mixed types. Specify dtype op
tion on import or set low_memory=False.
  if self.run_code(code, result):
```

```
Out[4]: Index(['community area', 'completion date', 'creation date',
               'current activity', 'latitude', 'location', 'longitude',
               'most recent action', 'number of potholes filled on block',
               'police district', 'service request number', 'service request ty
        pe',
               'ssa', 'status', 'street address', 'type of service request', 'w
        ard',
               'what is the nature of this code violation?',
               'what type of surface is the graffiti on?',
               'where is the graffiti located?', 'x coordinate', 'y coordinat
        e',
               'zip code', 'response time', 'community area number',
               'community area name', 'percent of housing crowded',
               'percent households below poverty', 'percent aged 16+ unemploye
        d',
               'percent aged 25+ without high school diploma',
               'percent aged under 18 or over 64', 'per capita income ',
               'hardship index', 'year'],
              dtype='object')
```

```
In [5]: # Taking a sample for speed
        df = df.sample(1000)
        df.head(1)
```

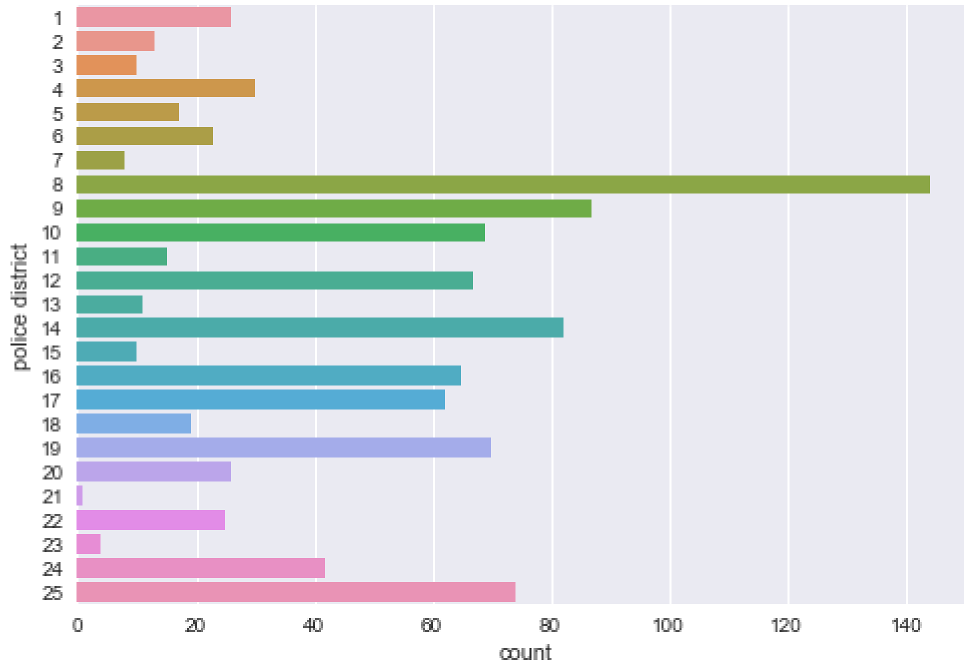
Out[5]:

	community area	completion date	creation date	current activity	latitude	location	long
1309183	41	2014-03-11	2014-02-24	NaN	41.800788	(41.80078781134253, -87.59682045090932)	-87.

1 rows × 34 columns

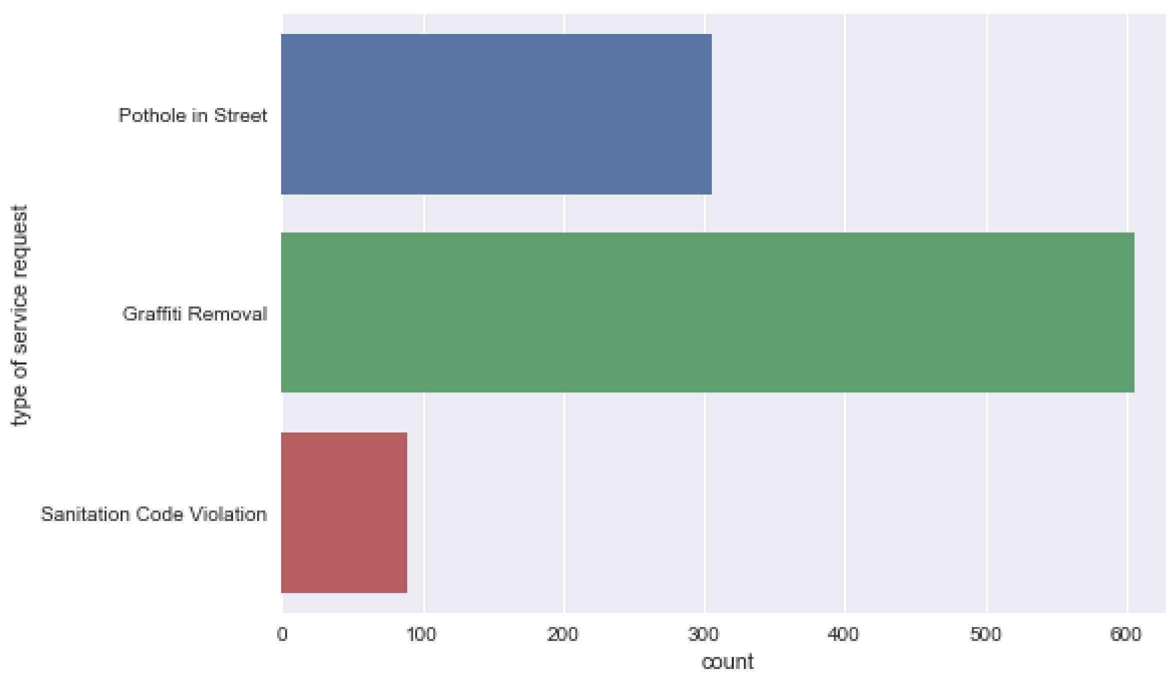
311 Calls by police district

In [6]:
```
ax = sns.countplot(y="police district", data=df)
```



311 Requests by Type

In [7]:
```
ax = sns.countplot(y="type of service request", data=df)
```



Top Ten Community Areas by Request Count

In [8]:
```
df['community area name'].value_counts().head(10)
```

Out[8]:
```
West Town          51
South Lawndale     49
Logan Square       45
Belmont Cragin     39
Avondale           33
Lower West Side    32
Portage Park       30
Brighton Park      30
Lake View          28
Chicago Lawn       28
Name: community area name, dtype: int64
```

In [9]:
```
# building a community area variable dataframe
indicators = df[['community area name','percent households below povert
y', 'percent aged 16+ unemployed',
       'percent aged 25+ without high school diploma',
       'percent aged under 18 or over 64', 'per capita income ',
       'hardship index']].groupby('community area name').max()

indicators.join(df['community area name'].value_counts())

# Adding response time
response_time = df.groupby('community area name')['response
time'].mean()
indicators = indicators.join(response_time)

# Adding request counts
indicators['total_requests'] = 0

request_types = df['type of service request'].unique()

for request_type in request_types:
    t = df[df['type of service request'] == request_type]
    t = t['community area name'].value_counts()
    t = t.rename('num' + request_type)
    indicators = indicators.join(t)
    indicators[t.name].fillna(0, inplace=True)
    indicators['total_requests'] = indicators['total_requests'] + indica
tors[t.name]
```

In [10]:
```
indicators
```

Out[10]:

	percent households below poverty	percent aged 16+ unemployed	percent aged 25+ without high school diploma	percent aged under 18 or over 64	per capita income	hardship index	response time
community area name							
Albany Park	19.2	10.0	32.9	32.0	21323	53.0	7.000000
Archer Heights	14.1	16.5	35.9	39.2	16134	67.0	4.363636
Armour Square	40.1	16.7	34.5	38.3	16148	82.0	8.200000
Ashburn	10.4	11.7	17.7	36.9	23482	37.0	37.230769
Auburn Gresham	27.6	28.3	18.5	41.9	15528	74.0	17.666667
Austin	28.6	22.6	24.4	37.9	15957	73.0	15.625000
Avalon Park	17.2	21.1	10.6	39.3	24454	41.0	3.000000
Avondale	15.3	9.2	24.7	31.0	20039	42.0	18.969697
Belmont Cragin	18.7	14.6	37.3	37.3	15461	70.0	16.025641
Beverly	5.1	8.0	3.7	40.5	39523	12.0	21.500000
Bridgeport	18.9	13.7	22.2	31.3	22694	43.0	3.428571
Brighton Park	23.6	13.9	45.1	39.3	13089	84.0	6.433333
Burnside	33.0	18.6	19.3	42.7	12515	79.0	75.000000
Calumet Heights	11.5	20.0	11.0	44.0	28887	38.0	15.000000
Chatham	27.8	24.0	14.5	40.3	18881	60.0	13.400000
Chicago Lawn	27.9	17.1	31.2	40.6	13231	80.0	4.107143
Clearing	8.9	9.5	18.8	37.6	25113	29.0	3.625000
Douglas	29.6	18.2	14.3	30.7	23791	47.0	0.000000
Dunning	10.6	10.0	16.2	33.6	26282	28.0	52.000000

	percent households below poverty	percent aged 16+ unemployed	percent aged 25+ without high school diploma	percent aged under 18 or over 64	per capita income	hardship index	response time
community area name							
East Garfield Park	42.4	19.6	21.3	43.2	12961	83.0	11.000000
East Side	19.2	12.1	31.9	42.8	17104	64.0	6.428571
Edgewater	18.2	9.2	9.7	23.8	33385	19.0	13.470588
Edison Park	3.3	6.5	7.4	35.3	40959	8.0	1.250000
Englewood	46.6	28.0	28.5	42.5	11888	94.0	39.666667
Forest Glen	7.5	6.8	4.9	40.5	44164	11.0	17.750000
Fuller Park	51.2	33.9	26.6	44.9	10432	97.0	1.000000
Gage Park	23.4	18.2	51.5	38.8	12171	93.0	24.076923
Garfield Ridge	8.8	11.3	19.3	38.1	26353	32.0	5.666667
Grand Boulevard	29.3	24.3	15.9	39.5	23472	57.0	7.000000
Greater Grand Crossing	29.6	23.0	16.5	41.0	17285	66.0	8.428571
...	...	...	...	...	...	...	...
Mount Greenwood	3.4	8.7	4.3	36.8	34381	16.0	9.333333
Near North Side	12.9	7.0	2.5	22.6	88669	1.0	5.727273
Near South Side	13.8	4.9	7.4	21.8	59077	7.0	202.000000
Near West Side	20.6	10.7	9.6	22.2	44689	15.0	8.173913
New City	29.0	23.0	41.5	38.9	12765	91.0	29.684211
North Center	7.5	5.2	4.5	26.2	57123	6.0	11.611111

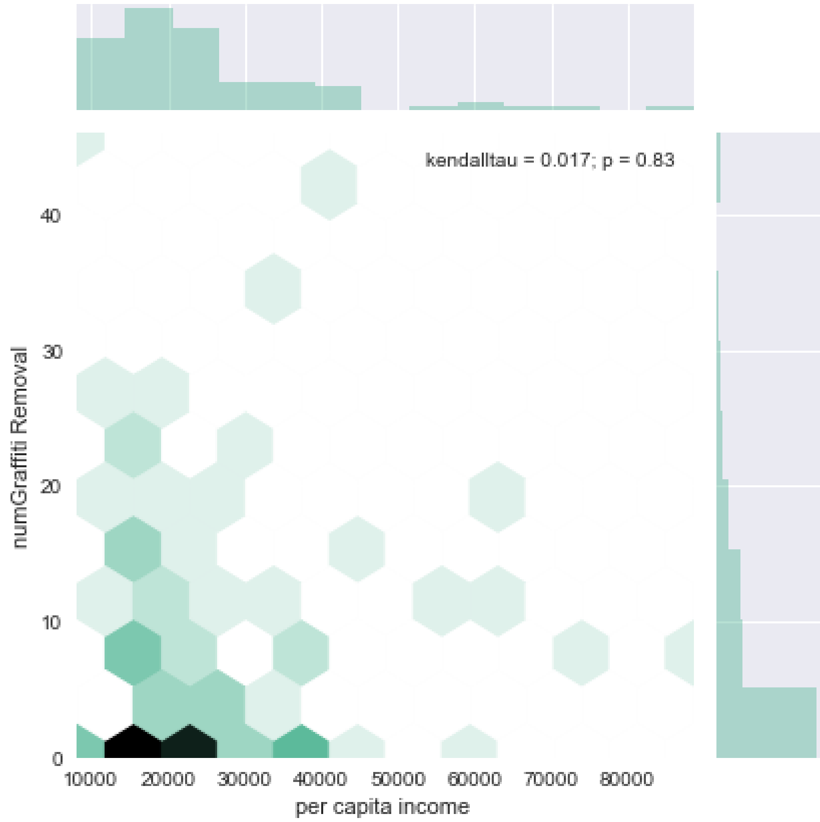
	percent households below poverty	percent aged 16+ unemployed	percent aged 25+ without high school diploma	percent aged under 18 or over 64	per capita income	hardship index	response time
community area name							
North Lawndale	43.1	21.2	27.6	42.7	12034	87.0	8.428571
North Park	13.2	9.9	14.4	39.0	26576	33.0	1.750000
Norwood Park	5.4	9.0	11.5	39.5	32875	21.0	21.400000
O'Hare	15.4	7.1	10.9	30.3	25828	24.0	16.000000
Portage Park	11.6	12.6	19.3	34.0	24336	35.0	16.900000
Pullman	21.6	22.8	13.1	38.6	20588	51.0	5.000000
Riverdale	56.5	34.6	27.5	51.5	8201	98.0	0.000000
Rogers Park	23.6	8.7	18.2	27.5	23939	39.0	3.076923
Roseland	19.8	20.3	16.9	41.2	17949	52.0	5.363636
South Chicago	29.8	19.7	26.6	41.1	16579	75.0	10.500000
South Deering	29.2	16.3	21.0	39.5	14685	65.0	21.333333
South Lawndale	30.7	15.8	54.8	33.8	10402	96.0	7.571429
South Shore	31.1	20.0	14.0	35.7	19398	55.0	16.750000
Uptown	24.0	8.9	11.8	22.2	35787	20.0	11.823529
Washington Height	16.9	20.8	13.7	42.6	19713	48.0	25.666667
Washington Park	42.1	28.6	25.4	42.8	13785	88.0	8.000000
West Elsdon	15.6	16.7	37.0	37.7	15754	69.0	33.769231
West Englewood	34.4	35.9	26.3	40.7	11317	89.0	3.800000

	percent households below poverty	percent aged 16+ unemployed	percent aged 25+ without high school diploma	percent aged under 18 or over 64	per capita income	hardship index	response time	t
community area name								
West Garfield Park	41.7	25.8	24.5	43.6	10934	92.0	19.000000	4
West Lawn	14.9	9.6	33.6	39.6	16907	56.0	3.900000	2
West Pullman	25.9	19.4	20.5	42.1	16563	62.0	9.571429	7
West Ridge	17.2	8.8	20.8	38.5	23040	46.0	20.538462	2
West Town	14.7	6.6	12.9	21.7	43198	10.0	7.352941	5
Woodlawn	30.7	23.4	16.5	36.1	18672	58.0	9.000000	3

76 rows × 11 columns

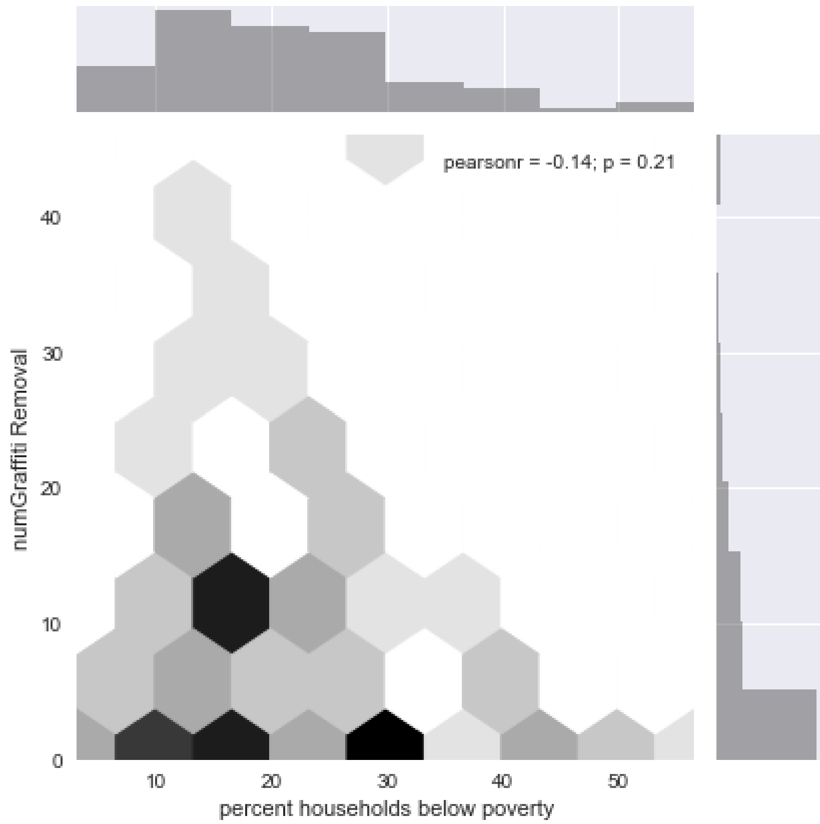
Graffiti locations

In [11]:
```
from scipy.stats import kendalltau
x = indicators['per capita income ']
y = indicators['numGraffiti Removal']
ax = sns.jointplot(x, y, kind="hex", stat_func=kendalltau, color="#4CB39
1")
```



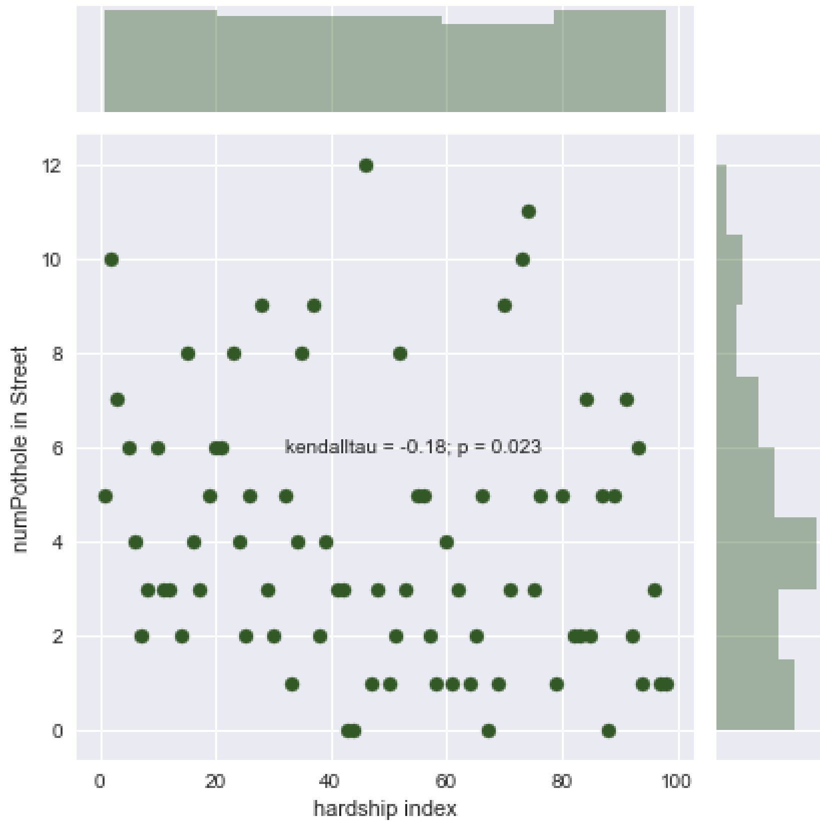
In [ ]:
```
### Poverty and Graffiti
```

In [12]:
```
x = indicators['percent households below poverty']
y = indicators['numGraffiti Removal']
ax = sns.jointplot(x, y, kind="hex", color="#333333")
```



Hardship and Potholes

In [13]:
```
x = indicators['hardship index']
y = indicators['numPothole in Street']
ax = sns.jointplot(x, y, stat_func=kendalltau, color="#325925")
```

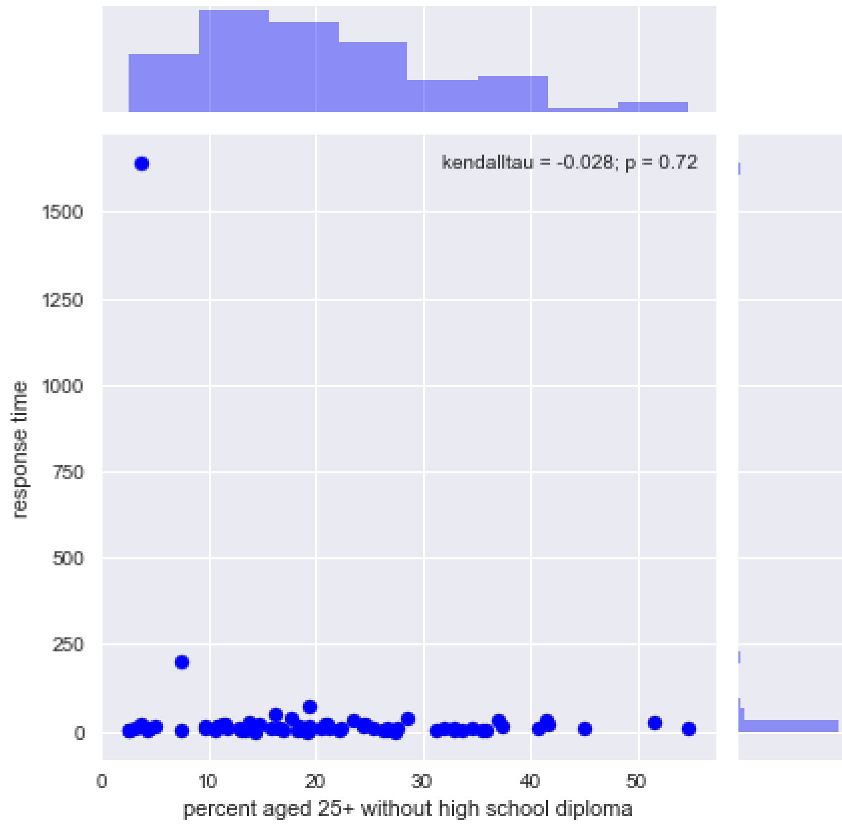


Avg Response Time by Community Area (top 10)

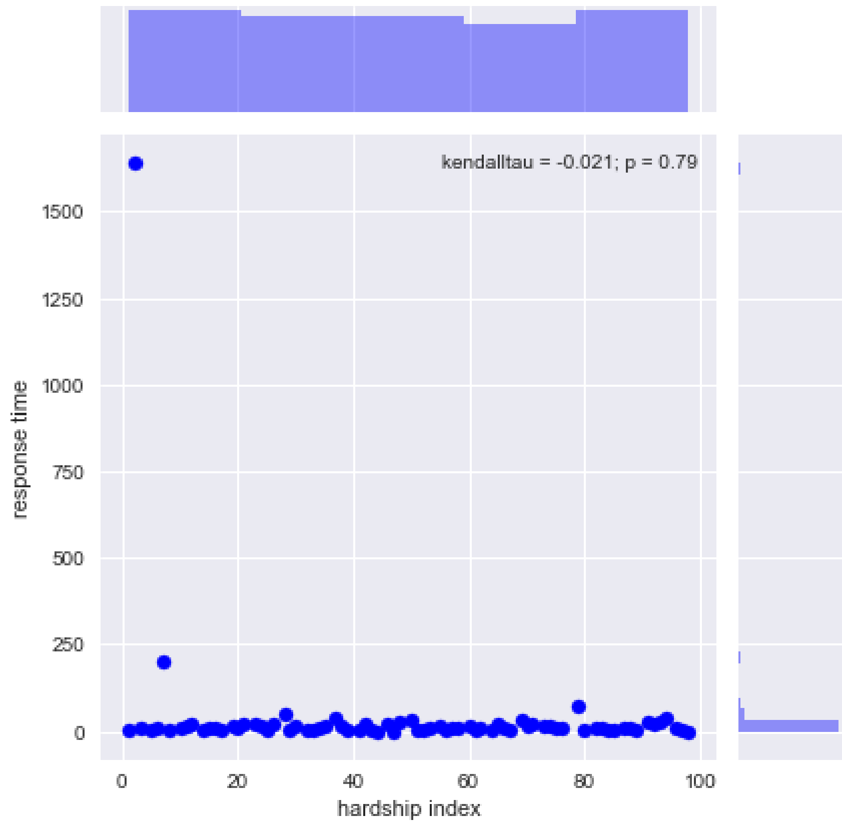
In [14]:
```
indicators['response time'].sort_values(ascending=False).head(10)
```

Out[14]:
```
community area name
Lincoln Park          1639.368421
Near South Side        202.000000
Burnside                75.000000
Dunning                 52.000000
Englewood               39.666667
Ashburn                 37.230769
West Elsdon             33.769231
Montclaire              31.000000
New City                29.684211
Washington Height       25.666667
Name: response time, dtype: float64
```

In [15]:
```
x = indicators['percent aged 25+ without high school diploma']
y = indicators['response time']
ax = sns.jointplot(x, y, stat_func=kendalltau, color="blue")
```



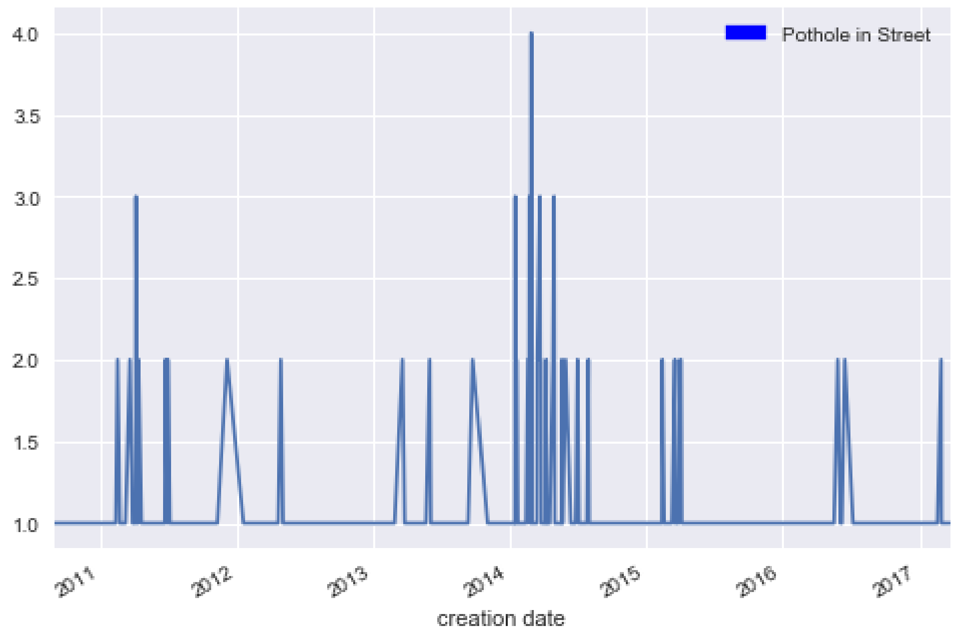
In [16]:
```
x = indicators['hardship index']
y = indicators['response time']
ax = sns.jointplot(x, y, stat_func=kendalltau, color="blue")
```



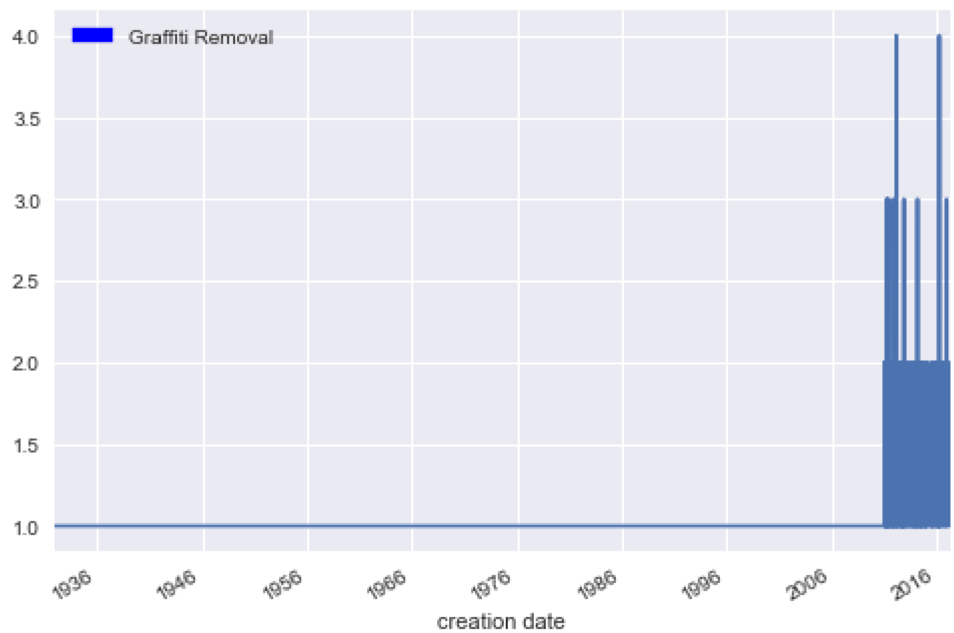
Types over time

In [17]:

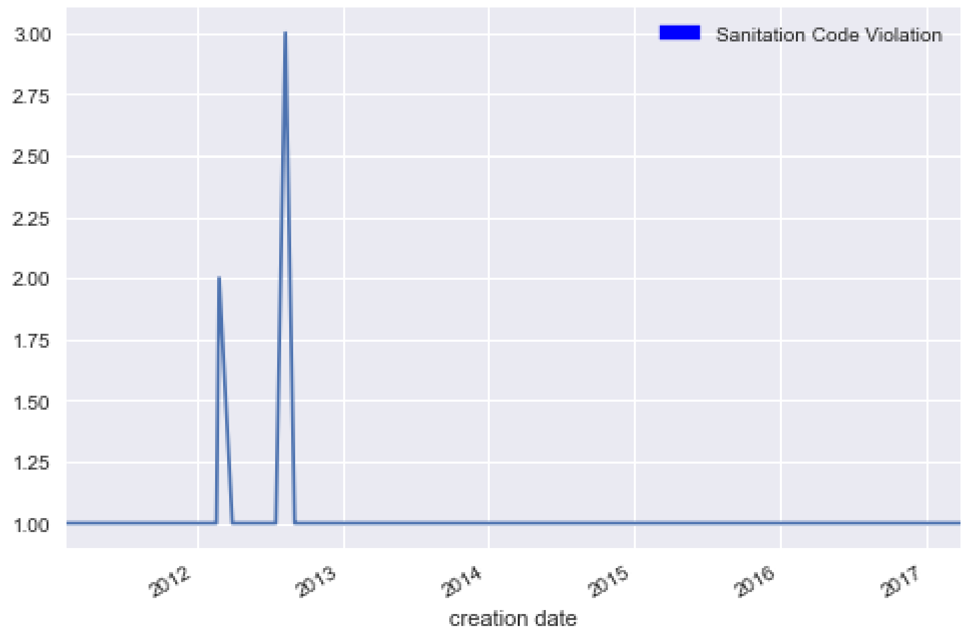
```
import matplotlib.patches as mpatches
import matplotlib.pyplot as plt
for t in df['type of service request'].unique():
    ax = df[df["type of service request"] == t][['creation date', "type
 of service request"]].groupby('creation date').count().plot()
    p = mpatches.Patch(color='blue', label=t)
    plt.legend(handles=[p])
plt.show()
```



Pothole in Street
4.0
3.5
3.0
2.5
2.0
1.5
1.0
2011
2012
2013
2014
2015
2016
2017
creation date



Graffiti Removal
4.0
3.5
3.0
2.5
2.0
1.5
1.0
1936
1946
1956
1966
1976
1986
1996
2006
2016
creation date



Summary of findings

- Graffiti is the most frequent service request. Graffiti removal has been consistently reported in the data we've analyzed.
- Graffiti request tend to be in lower income areas, either by per capita income or households in poverty
- There doesn't appear to be any strong linear relationship between hardship index and potholes
- Potholes were an mostly an issue in 2014. They are reported less currently. Sanitation and Abandon Buildings were not reported often
- There doesn't appear to be any strong linear relationship between hardship or adults without a HS diploma and response time

Problem 2 - Data Augmentation with ACS

In [18]:
```
# storing
old = df
```

In [19]:
```
import requests
from requests.auth import HTTPBasicAuth

API_KEY = '9032d3c94c7f4afe905da54f889af02a6b51f63f'
request_url = "http://citysdk.commerce.gov"

variables = ['income','population','poverty','median_contract_rent','edu
cation_bachelors','poverty_family']

def acs_data (zip_code):
    request_obj = {
        'level': 'tract',
        'zip': int(zip_code),
        'sublevel': False,
        'api': 'acs5',
        'year': 2014,
        'variables': variables
        }

    data = None

    response = requests.post(request_url, auth=HTTPBasicAuth(API_KEY, No
ne), json=request_obj)
    if response:
        data = response.json()
        data = data['features'][0]['properties']

    return data
```

In [20]:
```
# pull data by from zip code
codes = df[['zip code']]
codes = codes.drop_duplicates()

acs = codes.apply(acs_data, axis=1)
acs = acs.apply(pd.Series)
```

In [21]:
```
acs_ = acs
# humanize
columns = ['B01003_001E', 'B15003_022E', 'B17001_002E','B17012_002E', 'B
19013_001E', 'B25058_001E']
ref = {"B15003_022E": "education_bachelors",
        "B19013_001E":   "income",
        "B25058_001E":   "median_contract_rent",
        "B01003_001E":   "population",
        "B17001_002E":   "poverty",
        "B17012_002E":   "poverty_family"}
new_name = [ref[key] for key in columns]
acs = acs[columns]
acs.columns = new_name
```

In [22]:
```
# combine zip with acs
acs = pd.concat([acs,codes], axis=1)
```

In [23]:
```
# combine df with acs on zip
df = df.merge(acs, on='zip code')
```

In [48]:
```
# combine all indicators
new_indicators = df.groupby('community area name').max()
new_indicators = new_indicators.merge(indicators)
```

Type by Community Area

In [ ]:
```
g = sns.FacetGrid(df, col="type of service request")
ax = g.map(plt.hist, "community area")
```

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

I don't have any data in my sample on these

What types of blocks get “Sanitation Code Complaints”?

community area 47, near the Chatam neighborhood, which has lower incomes and 18% unemployment

Does that change over time in the data you collected?

There is a consitent bimodal distribution over the years, but it has changed slightly.

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

I don't have any data in my sample on these

In [ ]:
```
###
```

In [49]:
```
df['type of service request'].unique()
```

Out[49]:
```
array(['Pothole in Street', 'Graffiti Removal', 'Sanitation Code Violat
ion'], dtype=object)
```

In [45]:
```
new_indicators[new_indicators['community area'] == 47]
```

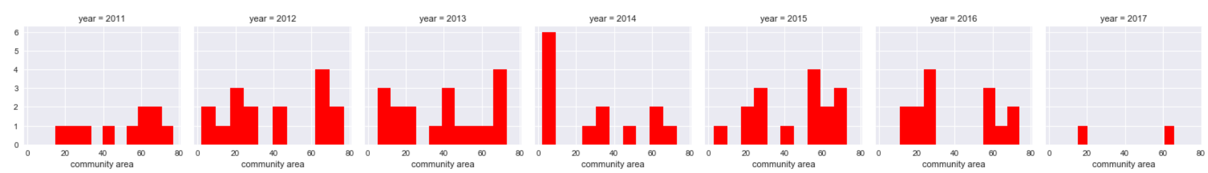
Out[45]:

	community area	completion date	creation date	latitude	location	longitude	number of potholes filled on block
0	47	2014-08-02	2014-05-19	41.734707	(41.73470701625012, -87.59735991511906)	-87.59736	30.0

1 rows × 32 columns

Sanitation Code Violations over time

In [51]:
```
data = df[df['type of service request'] == 'Sanitation Code Violation']
g = sns.FacetGrid(data, col="year")
g = g.map(plt.hist, "community area", color="r")
```



Problem 3

Assume you are running the 311 call center for Chicago. You get a call from 7500 S Wolcott Ave.

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? 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? 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 [ ]:
```
total = indicators['numGraffiti Removal'].sum()
eng_prob = (indicators[indicators.index == 'Englewood']['numGraffiti Removal'] / total).values
lawn_prob = (indicators[indicators.index == 'North Lawndale']['numGraffiti Removal'] / total).values
print("engleside prob = ", eng_prob)
print("lawndale prob = ", lawn_prob)
```

Answer:

Given the sample data, they have an equal chance of happening.

With the synthetic data, the prob a call is 60% and 40% for Englewood and Uptown.

- Prob(Englewood | Graffiti) = 100/600 x 600/1000 = 10%
- Prob(Uptown | Graffiti) = 160/400 x 400/1000 = 16%