ChicagoHomicdes

November 10, 2019

[1]:

```python
## Import Packages
import numpy as np  # useful for many scientific computing in Python
import pandas as pd # primary data structure library
import random # library for random number generation
import matplotlib.pyplot as plt # plotting library
# backend for rendering plots within the browser
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn.datasets.samples_generator import make_blobs
!conda install -c conda-forge folium=0.5.0 --yes
import folium
!pip install git+git://github.com/Toblerity/Shapely.git
print('Libraries imported.')
```

```
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.11
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda



# All requested packages already installed.

Collecting git+git://github.com/Toblerity/Shapely.git
  Cloning git://github.com/Toblerity/Shapely.git to /tmp/pip-req-build-6dipl1sq
  Running command git clone -q git://github.com/Toblerity/Shapely.git /tmp/pip-
req-build-6dipl1sq
Building wheels for collected packages: Shapely
  Building wheel for Shapely (setup.py) … done
  Stored in directory: /tmp/pip-ephem-wheel-cache-a182y314/wheels/42/0f/aa
/cfbde7df67ccee15095af37845476a53852d931e324f0c7236
Successfully built Shapely
```

```
Installing collected packages: Shapely
Successfully installed Shapely-1.7a2
Libraries imported.
```

[2]:

```python
hom_df = pd.read_csv('Homicide_Map.csv')
hom_df.head()
```

[2]:

```
      ID Case Number                    Date                 Block  IUCR  \
0  24787    HN623995  09/20/2007 06:52:00 PM   015XX S SANGAMON ST   110   
1  21058    HW431623  08/31/2013 11:19:00 AM      117XX S HALE AVE   110   
2   1710    HH609795  08/28/2002 08:30:00 AM    030XX S WABASH AVE   110   
3  24780    JC449748  10/02/2019 02:56:00 PM  030XX S ST LOUIS AVE   110   
4  24776    JC456756  10/02/2019 09:15:00 AM    018XX S HAMLIN AVE   110   

  Primary Type          Description Location Description  Arrest  Domestic  \
0     HOMICIDE  FIRST DEGREE MURDER            APARTMENT    True     False   
1     HOMICIDE  FIRST DEGREE MURDER                 YARD    True     False   
2     HOMICIDE  FIRST DEGREE MURDER          SCHOOL YARD    True     False   
3     HOMICIDE  FIRST DEGREE MURDER                 AUTO   False     False   
4     HOMICIDE  FIRST DEGREE MURDER               STREET   False     False   

   Beat  Ward FBI Code  X Coordinate  Y Coordinate  Year  \
0  1232  11.0      01A           NaN           NaN  2007   
1  2212  34.0      01A     1164670.0     1826809.0  2013   
2   133   3.0      01A     1177222.0     1884824.0  2002   
3  1032  22.0      01A     1153517.0     1884183.0  2019   
4  1014  24.0      01A     1151345.0     1890735.0  2019   

               Updated On   Latitude  Longitude                       Location  
0  10/10/2019 04:20:40 PM        NaN        NaN                            NaN  
1  10/10/2019 04:11:30 PM  41.680366 -87.672864  (41.680366299, -87.672864196)  
2  10/10/2019 04:11:30 PM  41.839293 -87.625170  (41.839292845, -87.625169769)  
3  10/09/2019 04:25:23 PM  41.838037 -87.712173  (41.838037246, -87.712173487)  
4  10/09/2019 04:25:23 PM  41.856060 -87.719972  (41.856059573, -87.719971925)  
```

[3]:

```python
#Remove Most of the Columns
df = hom_df.drop(['X Coordinate','Y Coordinate','Location','FBI Code','Case␣
 ↪Number','Updated On','Ward'], axis=1)
df.head()
```

[3]:

```
      ID                    Date                Block  IUCR Primary Type  \
0  24787  09/20/2007 06:52:00 PM   015XX S SANGAMON ST   110     HOMICIDE   
1  21058  08/31/2013 11:19:00 AM      117XX S HALE AVE   110     HOMICIDE   
2   1710  08/28/2002 08:30:00 AM    030XX S WABASH AVE   110     HOMICIDE   
3  24780  10/02/2019 02:56:00 PM  030XX S ST LOUIS AVE   110     HOMICIDE   
4  24776  10/02/2019 09:15:00 AM    018XX S HAMLIN AVE   110     HOMICIDE   
```

```
           Description Location Description  Arrest  Domestic  Beat  Year  \
0  FIRST DEGREE MURDER            APARTMENT    True     False  1232  2007
1  FIRST DEGREE MURDER                 YARD    True     False  2212  2013
2  FIRST DEGREE MURDER          SCHOOL YARD    True     False   133  2002
3  FIRST DEGREE MURDER                 AUTO   False     False  1032  2019
4  FIRST DEGREE MURDER               STREET   False     False  1014  2019

    Latitude  Longitude
0        NaN        NaN
1  41.680366 -87.672864
2  41.839293 -87.625170
3  41.838037 -87.712173
4  41.856060 -87.719972
```

[4]:

```
#Drop Unknown Lat/Lon
df1=df.dropna(axis=0)
df1.head()
```

[4]:

```
      ID                    Date                 Block  IUCR Primary Type  \
1  21058  08/31/2013 11:19:00 AM      117XX S HALE AVE   110      HOMICIDE
2   1710  08/28/2002 08:30:00 AM    030XX S WABASH AVE   110      HOMICIDE
3  24780  10/02/2019 02:56:00 PM  030XX S ST LOUIS AVE   110      HOMICIDE
4  24776  10/02/2019 09:15:00 AM    018XX S HAMLIN AVE   110      HOMICIDE
5  23534  08/20/2017 05:35:00 AM   077XX N ASHLAND AVE   110      HOMICIDE

           Description Location Description  Arrest  Domestic  Beat  Year  \
1  FIRST DEGREE MURDER                 YARD    True     False  2212  2013
2  FIRST DEGREE MURDER          SCHOOL YARD    True     False   133  2002
3  FIRST DEGREE MURDER                 AUTO   False     False  1032  2019
4  FIRST DEGREE MURDER               STREET   False     False  1014  2019
5  FIRST DEGREE MURDER            STAIRWELL    True     False  2422  2017

    Latitude  Longitude
1  41.680366 -87.672864
2  41.839293 -87.625170
3  41.838037 -87.712173
4  41.856060 -87.719972
5  42.021638 -87.670717
```

[5]:

```
#Cluster by Latitude and Longitude
LatLondf=df1.drop(['ID','Date','Block','Domestic','IUCR','Primary␣
 ↪Type','Description','Location Description','Arrest','Beat','Year'],axis=1)
LatLondf.head()
```

[5]:

```
    Latitude  Longitude
1  41.680366 -87.672864
2  41.839293 -87.625170
```

```
3  41.838037 -87.712173
4  41.856060 -87.719972
5  42.021638 -87.670717
```

[6]:

```
num_clusters =12

k_means = KMeans(init="k-means++", n_clusters=num_clusters, n_init=12)
k_means.fit(LatLondf)
labels = k_means.labels_
print(labels)
```

```
[3 0 8 … 8 4 4]
```

[7]:

```
LatLondf["Labels"] = labels
LatLondf.head(15)
```

[7]:

```
     Latitude  Longitude  Labels
1   41.680366 -87.672864       3
2   41.839293 -87.625170       0
3   41.838037 -87.712173       8
4   41.856060 -87.719972       8
5   42.021638 -87.670717       7
6   41.998438 -87.692343       7
7   41.880282 -87.762241       2
8   41.711715 -87.642954       3
9   41.765619 -87.580586      11
10  41.894496 -87.745792       2
11  41.704305 -87.634307       3
12  41.880994 -87.752640       2
13  41.766544 -87.628192      11
14  41.743852 -87.562464       5
15  41.764425 -87.628138      11
```

[8]:

```
LatLondf.groupby('Labels').mean()
```

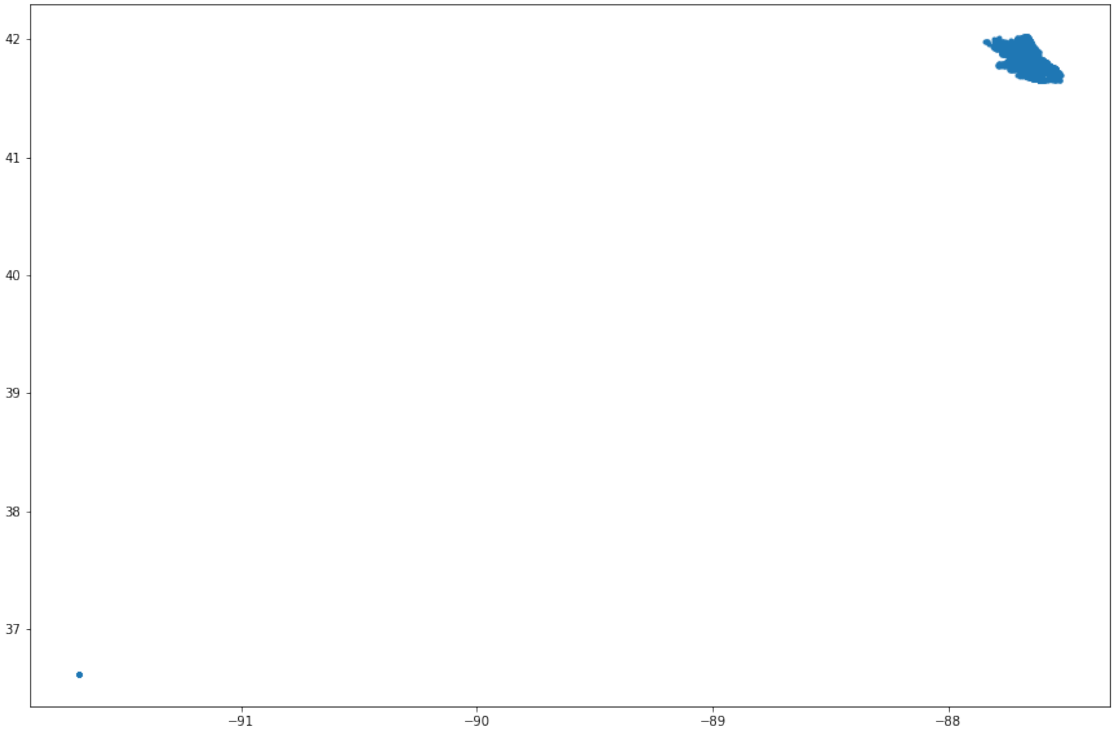
[8]:

```
         Latitude  Longitude
Labels
0       41.804048 -87.635544
1       36.619446 -91.686566
2       41.901364 -87.760368
3       41.690041 -87.629538
4       41.911017 -87.715490
5       41.740991 -87.562577
6       41.791147 -87.698265
7       41.980917 -87.679242
8       41.864763 -87.717392
9       41.757314 -87.657711
```

```
10      41.877233 -87.656014
11      41.759845 -87.605234
```

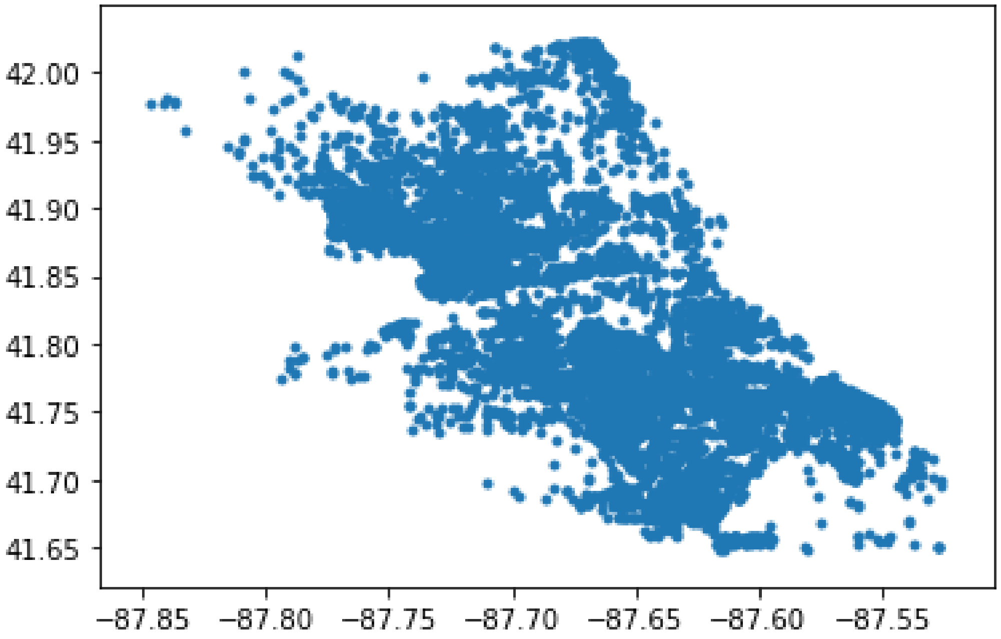
[9]:
```
##ScatterPlot
plt.figure(figsize=(15, 10))
plt.scatter(LatLondf['Longitude'], LatLondf['Latitude'], marker='.')
```

[9]:
```
<matplotlib.collections.PathCollection at 0x7fea794dfdd8>
```



[10]:
```
#Remove Outlier
LatLondf=LatLondf[LatLondf.Latitude > 41]
plt.scatter(LatLondf['Longitude'], LatLondf['Latitude'], marker='.')
```

[10]:
```
<matplotlib.collections.PathCollection at 0x7fea7946a940>
```



```
[11]: ncl=12 #number of clusters
      # Fit the k means mode
      k_means = KMeans(init="k-means++", n_clusters=ncl, n_init=ncl)
      k_means.fit(LatLondf)
```

```
[11]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
          n_clusters=12, n_init=12, n_jobs=None, precompute_distances='auto',
          random_state=None, tol=0.0001, verbose=0)
```

```
[12]: #Grab Labels
      k_means_labels = k_means.labels_
      k_means_labels
```

```
[12]: array([2, 1, 0, …, 0, 7, 7], dtype=int32)
```

```
[13]: # Get coordinates of Cluster Centers
      k_means_cluster_centers = k_means.cluster_centers_
      k_means_cluster_centers
```

```
[13]: array([[ 4.18647626e+01, -8.77173921e+01,  8.00000000e+00],
             [ 4.18040477e+01, -8.76355443e+01, -5.32907052e-14],
             [ 4.16900406e+01, -8.76295383e+01,  3.00000000e+00],
             [ 4.17598450e+01, -8.76052339e+01,  1.10000000e+01],
             [ 4.17911465e+01, -8.76982655e+01,  6.00000000e+00],
             [ 4.19404264e+01, -8.77739189e+01,  2.00000000e+00],
             [ 4.17573139e+01, -8.76577106e+01,  9.00000000e+00],
             [ 4.19110170e+01, -8.77154899e+01,  4.00000000e+00],
             [ 4.17409910e+01, -8.75625771e+01,  5.00000000e+00],
             [ 4.19809166e+01, -8.76792424e+01,  7.00000000e+00],
             [ 4.18772325e+01, -8.76560143e+01,  1.00000000e+01],
             [ 4.18928786e+01, -8.77574247e+01,  2.00000000e+00]])
```

[14]:

```
## Add Cluster Label to Dataframe
LatLondf["Cluster"] = k_means_labels
LatLondf.head()
```

[14]:

```
    Latitude  Longitude  Labels  Cluster
1  41.680366 -87.672864       3        2
2  41.839293 -87.625170       0        1
3  41.838037 -87.712173       8        0
4  41.856060 -87.719972       8        0
5  42.021638 -87.670717       7        9
```

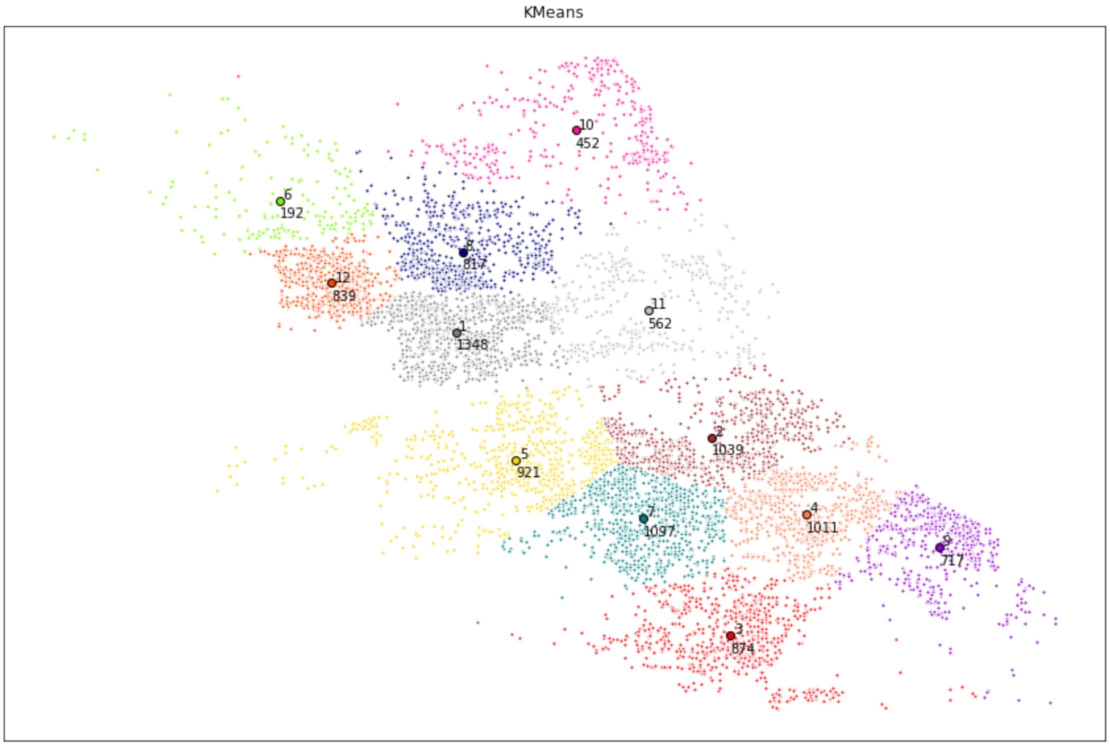
[15]:

```
## Find Number of Homicides in Each Cluster
countdf=LatLondf.groupby('Cluster').count()
print(countdf)
```

```
         Latitude  Longitude  Labels
Cluster
0            1348       1348    1348
1            1039       1039    1039
2             874        874     874
3            1011       1011    1011
4             921        921     921
5             192        192     192
6            1097       1097    1097
7             817        817     817
8             717        717     717
9             452        452     452
10            562        562     562
11            839        839     839
```

[27]:

```
##Make Plot
# initialize the plot with the specified dimensions.
fig = plt.figure(figsize=(15, 10))

# colors uses a color map, which will produce an array of colors based on
# the number of labels. We use set(k_means_labels) to get the
# unique labels.
colors = plt.cm.Spectral(np.linspace(0, 1, len(set(k_means_labels))))
cols=['gray','brown','red','coral','gold','lawngreen','teal','navy','darkviolet','deeppink','s
##not used:       'hot', 'afmhot', 'gist_heat', 'copper'
# create a plot
ax = fig.add_subplot(1, 1, 1)

# loop through the data and plot the datapoints and centroids.
# k will range from 0-3, which will match the number of clusters in the dataset.
for k, col in zip(range(len([[1,1], [-2, 2], [-3,␣
 ↪3],[-4,4],[-5,5],[-6,6],[-7,7],[-8,8],[-9,9],[-10,10],[-11,11],[-12,12]])),␣
 ↪colors):
#for k, col in zip(range([[4,4],[2,2],[-3,3],1,3],[1,2]]), colors):    # create␣
 ↪a list of all dapoints, where the datapoints that are
    # in the cluster (ex. cluster 0) are labeled as true, else they are
    # labeled as false.
    my_members = (k_means_labels == k)

    # define the centroid, or cluster center.
    cluster_center = k_means_cluster_centers[k]

    # plot the datapoints with color col.
    ax.plot(LatLondf.Longitude[my_members], LatLondf.Latitude[my_members], 'w',␣
 ↪markerfacecolor=cols[k], marker='.')

    # plot the centroids with specified color, but with a darker outline␣
 ↪#changed col to str(k)
    ax.plot(cluster_center[1], cluster_center[0], 'o', markerfacecolor=cols[k],␣
 ↪ markeredgecolor='k', markersize=6)

    # plot cluster number
    plt.text(cluster_center[1]+.001, cluster_center[0]+.001,str(k+1))

    # plot count of homicides in cluster
    plt.text(cluster_center[1], cluster_center[0]-.01,countdf.Latitude[k])


# title of the plot
ax.set_title('KMeans')

# remove x-axis ticks
ax.set_xticks(())

# remove y-axis ticks
ax.set_yticks(())

# show the plot
plt.show()
```



[17]:

```
!pip install geopy
from geopy.geocoders import Nominatim # convert an address into latitude and␣
 ↪longitude values
address = 'Chicago, IL'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Chicago are {}, {}.'.format(latitude,␣
 ↪longitude))
```

```
Collecting geopy
  Downloading https://files.pythonhosted.org/packages/80/93/d384479da0ead7
12bdaf697a8399c13a9a89bd856ada5a27d462fb45e47b/geopy-1.20.0-py2.py3-none-any.whl
(100kB)
     |                 | 102kB 19.0MB/s ta 0:00:01
Collecting geographiclib<2,>=1.49 (from geopy)
  Downloading https://files.pythonhosted.org/packages/8b/62/26ec95a98ba642991631
99e95ad1b0e34ad3f4e176e221c40245f211e425/geographiclib-1.50-py3-none-any.whl
Installing collected packages: geographiclib, geopy
Successfully installed geographiclib-1.50 geopy-1.20.0
The geograpical coordinate of Chicago are 41.8755616, -87.6244212.
```

[30]:

```
##Get Libraries
import json # library to handle JSON files
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas␣
 ↪dataframe
#!conda install -c conda-forge folium=0.5.0 --yes # uncomment this line if you␣
 ↪haven't completed the Foursquare API lab
import folium # map rendering library
print('Libraries imported.')
```

```
Libraries imported.
```

[19]:

```
# create map of Chicago using latitude and longitude values
map_ChicagoBeats = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers to map
#for lat, lng in zip(LatLondf['Latitude'],␣
 ↪LatLondf['Longitude'],parse_html=False).add_to(map_ChicagoBeats)

map_ChicagoBeats
```

[19]:

```
<folium.folium.Map at 0x7f5d7a31b470>
```

[31]:

```
with open('BeatBound.geojson') as json_data:
    chicagobeat_data = json.load(json_data)
```

[32]:

```
beat_data=chicagobeat_data['features']
```

[33]:

```
len(beat_data)
```

[33]:

```
277
```

[23]:

```
#install more packages
from shapely.geometry import Point, Polygon
!pip install geojson
!pip install geojsonio
from matplotlib import pyplot as plt
from matplotlib.patches import Circle, Wedge, Polygon

!pip install descartes
from descartes import PolygonPatch
import math
#import urllib2
!pip install simplejson
import simplejson
```

```
Collecting geojson
```

```
  Downloading https://files.pythonhosted.org/packages/e4/8d/9e28e9af95739e6d2d2f
8d4bef0b3432da40b7c3588fbad4298c1be09e48/geojson-2.5.0-py2.py3-none-any.whl
Installing collected packages: geojson
Successfully installed geojson-2.5.0
Collecting geojsonio
  Downloading https://files.pythonhosted.org/packages/7f/42/a773a4d4a6a78261dce4
18269cd017d8ff401206bc5724d9390084ebbf3d/geojsonio-0.0.3.tar.gz
Collecting github3.py (from geojsonio)
  Downloading https://files.pythonhosted.org/packages/a6/21/9055a739fbe7b2
2a8e99e42906f2c75ba02bab9fd193a85837cd1d6e55d3/github3.py-1.3.0-py2.py3-none-
any.whl (153kB)
     |                 | 153kB 17.8MB/s eta 0:00:01
Requirement already satisfied: six in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from geojsonio)
(1.12.0)
Collecting uritemplate>=3.0.0 (from github3.py->geojsonio)
  Downloading https://files.pythonhosted.org/packages/e5/7d/9d5a640c4f8bf2c8b1af
c015e9a9d8de32e13c9016dcc4b0ec03481fb396/uritemplate-3.0.0-py2.py3-none-any.whl
Requirement already satisfied: requests>=2.18 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
github3.py->geojsonio) (2.22.0)
Requirement already satisfied: python-dateutil>=2.6.0 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
github3.py->geojsonio) (2.8.0)
Collecting jwcrypto>=0.5.0 (from github3.py->geojsonio)
  Downloading https://files.pythonhosted.org/packages/f0/0d/00173a6aee1025
e529b21c365182c8d06e78b1beb98d5633f841da6f122e/jwcrypto-0.6.0-py2.py3-none-
any.whl (73kB)
     |                 | 81kB 18.7MB/s eta 0:00:01
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.18->github3.py->geojsonio) (1.25.6)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.18->github3.py->geojsonio) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.18->github3.py->geojsonio) (2.8)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests>=2.18->github3.py->geojsonio) (2019.9.11)
Requirement already satisfied: cryptography>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
jwcrypto>=0.5.0->github3.py->geojsonio) (2.7)
Requirement already satisfied: cffi!=1.11.3,>=1.8 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
cryptography>=1.5->jwcrypto>=0.5.0->github3.py->geojsonio) (1.13.0)
Requirement already satisfied: asn1crypto>=0.21.0 in
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
cryptography>=1.5->jwcrypto>=0.5.0->github3.py->geojsonio) (1.2.0)
Requirement already satisfied: pycparser in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
cffi!=1.11.3,>=1.8->cryptography>=1.5->jwcrypto>=0.5.0->github3.py->geojsonio)
(2.19)
Building wheels for collected packages: geojsonio
  Building wheel for geojsonio (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/a9/ef/7c/7bbf228
825e8717adaa84cd4b6c4ed8649b7958dd2bac45076
Successfully built geojsonio
Installing collected packages: uritemplate, jwcrypto, github3.py, geojsonio
Successfully installed geojsonio-0.0.3 github3.py-1.3.0 jwcrypto-0.6.0
uritemplate-3.0.0
Collecting descartes
  Downloading https://files.pythonhosted.org/packages/e5/b6/1ed2eb03989ae5745846
64985367ba70cd9cf8b32ee8cad0e8aaeac819f3/descartes-1.1.0-py3-none-any.whl
Requirement already satisfied: matplotlib in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from descartes)
(3.1.1)
Requirement already satisfied: python-dateutil>=2.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
matplotlib->descartes) (2.8.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
matplotlib->descartes) (2.4.2)
Requirement already satisfied: cycler>=0.10 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
matplotlib->descartes) (0.10.0)
Requirement already satisfied: numpy>=1.11 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
matplotlib->descartes) (1.15.4)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
matplotlib->descartes) (1.1.0)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from python-
dateutil>=2.1->matplotlib->descartes) (1.12.0)
Requirement already satisfied: setuptools in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
kiwisolver>=1.0.1->matplotlib->descartes) (41.4.0)
Installing collected packages: descartes
Successfully installed descartes-1.1.0
Collecting simplejson
  Downloading https://files.pythonhosted.org/packages/e3/24/c35fb1c1c315fc
0fffe61ea00d3f88e85469004713dab488dee4f35b0aff/simplejson-3.16.0.tar.gz (81kB)
     |                 | 81kB 2.2MB/s eta 0:00:01
Building wheels for collected packages: simplejson
```

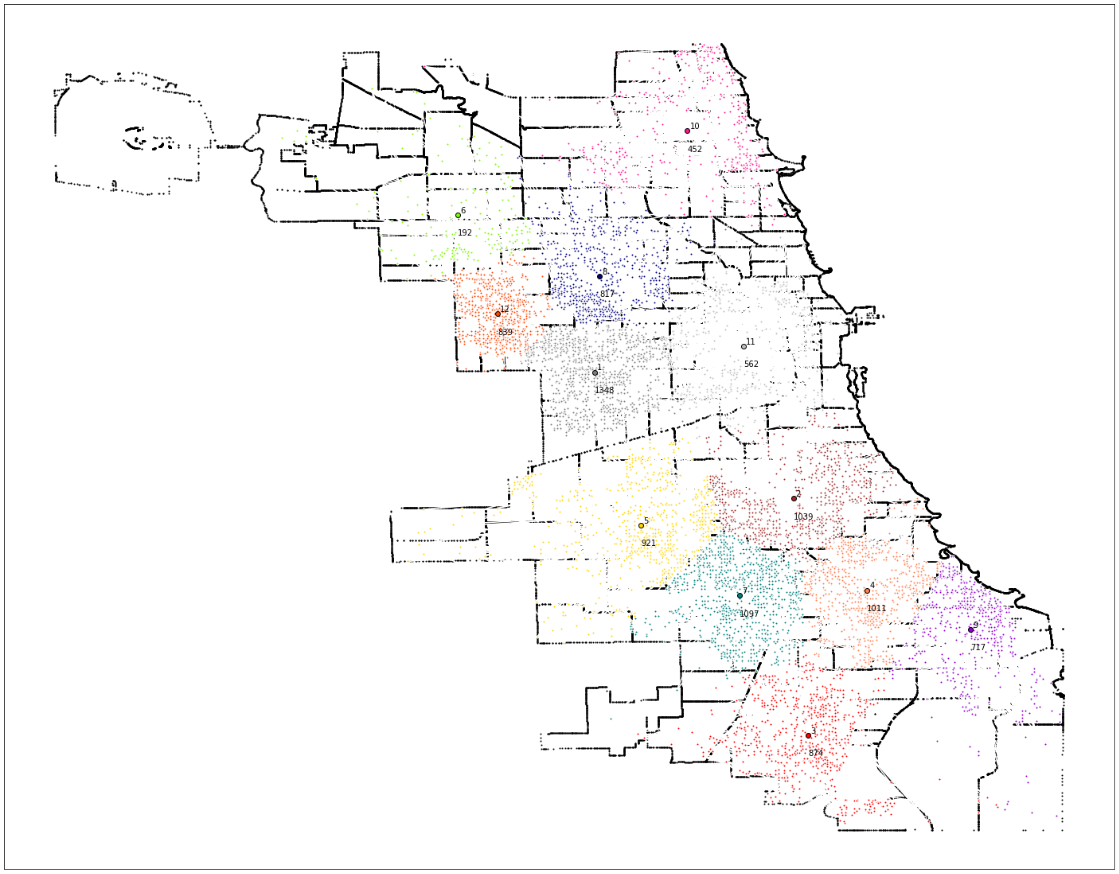
```
  Building wheel for simplejson (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/5d/1a/1e/0350bb3
df3e74215cd91325344cc86c2c691f5306eb4d22c77
Successfully built simplejson
Installing collected packages: simplejson
Successfully installed simplejson-3.16.0
```

[34]:

```
##Make Plot
# initialize the plot with the specified dimensions.
fig = plt.figure(figsize=(25, 20))



# create a plot
ax = fig.add_subplot(1, 1, 1)

# add beat boundaries to the plot
ind = 0;
while (ind < 277):
    for coordlist in beat_data[ind]['geometry']['coordinates']:
        data=np.array(coordlist)
        flat=[]
        for i in data:
            for j in i:
                flat.append(j)
        x,y=np.asarray(flat).T
        #plt.scatter(x,y,s=1.5,c='k',marker='o')
        ax.plot(x, y, 'o', markerfacecolor='k',  markeredgecolor='k',␣
 ↪markersize=1)
        ind += 1

cols=['gray','brown','red','coral','gold','lawngreen','teal','navy','darkviolet','deeppink','s
##not used:       'hot', 'afmhot', 'gist_heat', 'copper'
# create a plot
ax = fig.add_subplot(1, 1, 1)

# loop through the data and plot the datapoints and centroids.
# k will range from 0-3, which will match the number of clusters in the dataset.
for k, col in zip(range(len([[1,1], [-2, 2], [-3,␣
 ↪3],[-4,4],[-5,5],[-6,6],[-7,7],[-8,8],[-9,9],[-10,10],[-11,11],[-12,12]])),␣
 ↪colors):
#for k, col in zip(range([[4,4],[2,2],[-3,3],1,3],[1,2]]), colors):    # create␣
 ↪a list of all dapoints, where the datapoints that are
    # in the cluster (ex. cluster 0) are labeled as true, else they are
    # labeled as false.
    my_members = (k_means_labels == k)

    # define the centroid, or cluster center.
    cluster_center = k_means_cluster_centers[k]

    # plot the datapoints with color col.
    ax.plot(LatLondf.Longitude[my_members], LatLondf.Latitude[my_members], 'w',␣
 ↪markerfacecolor=cols[k], marker='.')

    # plot the centroids with specified color, but with a darker outline␣
 ↪#changed col to str(k)
    ax.plot(cluster_center[1], cluster_center[0], 'o', markerfacecolor=cols[k],␣
 ↪ markeredgecolor='k', markersize=6)

    # plot cluster number
    plt.text(cluster_center[1]+.001, cluster_center[0]+.001,str(k+1))

    # plot count of homicides in cluster
    plt.text(cluster_center[1], cluster_center[0]-.01,countdf.Latitude[k])


# title of the plot
ax.set_title('KMeans')

# remove x-axis ticks
ax.set_xticks(())

# remove y-axis ticks
ax.set_yticks(())

# show the plot
plt.show()
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:26: MatplotlibDeprecationWarning: Adding an axes
using the same arguments as a previous axes currently reuses the earlier
instance.  In a future version, a new instance will always be created and
returned.  Meanwhile, this warning can be suppressed, and the future behavior
ensured, by passing a unique label to each axes instance.
```

KMeans



```
[35]: plt.savefig('homicde_map.png')
```

```
<Figure size 432x288 with 0 Axes>
```

```
[20]: cols[0]
```

```
[20]: 'binary'
```

```
[21]: cols[1]
```

```
[21]: 'gist_yarg'
```

```
[ ]:
```