```
'''
bsb134: run linux, and run in terminal: /usr/local/anaconda/bin/spyder
talk about interface, change fonts, just have code and interpreter and drop
other windows; can just use Ipython, that's the trend
'''

'''
win setup
on windows: anaconda downloades easily from
https://www.anaconda.com/download/
get python 3.6 version
how to determine if 64 or 32 bit:
https://www.google.com/search?q=windows+64+or+32+bit

and
click windows search from windows icon at bottom left and find
'anaconda prompt' and run
conda install -c conda-forge geopandas
'''

#geopandas 0.3 also installs fine on apps.rutgers just may need --user option
pip.main(['install', 'geopandas','--user'])
pip.main(['install', 'pysal','--user'])

wd='path to project dir' #on win may need double like c:\\docs\\etc\\
#bsb134: get path by creating sth on desktop, right click and properties:
# like /crab/u12/facstaff/ao264/Desktop/lab2

import os
import urllib
os.getcwd()
os.makedirs(wd)
os.chdir(wd)
os.getcwd()
os.listdir()

%matplotlib #this will pop out graphs from ipython; run beofre matplotlib!
# and will go back %matplotlib inline

import matplotlib
import matplotlib.pyplot as plt

import geopandas as gpd



#LATER: WAS DOING SOME BASIC BASIC INTRO EXAMPLES BUT DIDNT FINISH, EITHER
#FINISH OR DUMP!

# world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))

# world.plot()

# world['gdp_per_cap'] = world.gdp_md_est / world.pop_est



# #per size guess just make map bigger and legen will be smaller duh
# #

# world.plot(column='pop_est',legend=True,cmap='YlOrRd',scheme='QUANTILES',k=3
#  ,figsize=(18,8) )

# #legend_kwds=(fontsize=20)

# #import pylab
# import matplotlib.pyplot as plt

# f, ax = plt.subplots(1, figsize=(10,6))
# world.plot(column='pop_est',legend=True,cmap='YlOrRd',scheme='QUANTILES',k=3,
#            edgecolor='grey',facecolor='blue',color='black',linewidth=2,ax=ax)
# #not sure what facelor or color is! guess one must show when poly empty eg missing
```

```
# #scheme 'Equal_interval', 'Quantiles', 'Fisher_Jenks'
# leg = ax.get_legend()
# #leg.set_bbox_to_anchor((1,1,0,0)) #top right
# #leg.set_bbox_to_anchor((0.3,1,0,0)) #top left
# leg.set_bbox_to_anchor((0.3,0.2,0,0)) #bottom left
# #ax.set_axis_off() #kills axes
# ax.set_xticks([]) #kills xticks
# ax.set_yticks([]) #kills xticks
# ax.set_title('Watersheds by area ($mi^2$)')
# f.savefig('foo.pdf') # can also have png etc extensions


# # cmap=plt.cm.Blues
# #plt.axis('equal') #not sure what it does!

# ##### points and overlays


# cities = gpd.read_file(gpd.datasets.get_path('naturalearth_cities'))
# cities.plot(marker='*', color='green', markersize=5)

# #check crs
# cities = cities.to_crs(world.crs)

# base = world.plot(color='white', edgecolor='black')
# cities.plot(ax=base, marker='o', color='red', markersize=5)


# f, ax = plt.subplots()
# # set aspect to equal. This is done automatically
# # when using *geopandas* plot on it's own, but not when
# # working with pyplot directly.
# ax.set_aspect('equal')
# world.plot(ax=ax, color='white', edgecolor='black')
# cities.plot(ax=ax, marker='o', color='red', markersize=5)
# #plt.show()




# #####merging
# In [1]: world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))


# country_shapes = world[['geometry', 'iso_a3']]
# country_names = world[['name', 'iso_a3']]

# TODO continue here




##### example: nj counties!

urllib.request.urlretrieve('https://njgin.state.nj.us/oit/gis/download/bounds_nj_shp.zip','nj-cou
nties.zip')

import zipfile
zip_ref = zipfile.ZipFile('nj-counties.zip', 'r')
zip_ref.extractall()
zip_ref.close()

njCounties=gpd.read_file('nj_counties.shp')

# have a look at the data and replace val
njCounties.head()
njCounties.dtypes
njCounties.describe().round(2)
njCounties[['COUNTY','POPDEN2010']]
```

```
njCounties['POPDEN2010'].replace(450,450.9,inplace=True) #just an example

njCounties[['COUNTY','POPDEN2010']]
njCounties['POPDEN2010'].hist(bins=5)
njCounties.boxplot(column='POPDEN2010',by='REGION'), plt.show()

fig, ax = plt.subplots()
ax.scatter(njCounties['POPDEN2010'],njCounties['POP2000'] )
plt.xlabel('den', fontsize=18)
plt.ylabel('pop', fontsize=16)

for i, txt in enumerate(njCounties['COUNTY']):
    ax.annotate(txt, (njCounties['POPDEN2010'][i],njCounties['POP2000'][i]),fontsize=8)


## and now off to mapping

njCounties.plot()
#plt.show()

f, ax = plt.subplots(1, figsize=(6,10))
njCounties.plot(column='POPDEN2010',legend=True,cmap='YlOrRd',scheme='QUANTILES',k=3, edgecolor='
grey',linewidth=2,ax=ax)
#scheme 'Equal_interval', 'Quantiles', 'Fisher_Jenks'
leg = ax.get_legend()
#leg.set_bbox_to_anchor((1,1,0,0)) #top right
#leg.set_bbox_to_anchor((0.3,1,0,0)) #top left
leg.set_bbox_to_anchor((0.3,0.2,0,0)) #bottom left
#ax.set_axis_off() #kills axes
ax.set_xticks([]) #kills xticks
ax.set_yticks([]) #kills xticks
ax.set_title('Watersheds by area ($mi^2$)')

urllib.request.urlretrieve('https://sites.google.com/site/adamokuliczkozaryn/gis_int/NJ-counties-
Zillow-Home-Value-Index-TimeSeries.xls','zillow.xls')

import pandas as pd
uglyZillow=pd.read_excel('zillow.xls')
uglyZillow.head() #ugly! clean up in excel!

urllib.request.urlretrieve('https://sites.google.com/site/adamokuliczkozaryn/gis_int/all_homes.cs
v','zillow.csv')
zillow=pd.read_csv('zillow.csv')
zillow.head()
zil1=zillow[['UPPER','Dec 2012']]
zil1.rename(columns={'UPPER': 'COUNTY'}, inplace=True)


#njCounties.dtypes
#njC1=njCounties[['COUNTY','POP2010']] #guess messes up shp

njC1 = njCounties.merge(zil1, on='COUNTY')

njC1.dtypes
njC1[['COUNTY','Dec 2012']]

#njC2=njC1.drop(njC1.index[[19]]) #specify after commas obs to drop
import numpy as np
njC2 = njC1[np.isfinite(njC1['Dec 2012'])] #drop missing on Dec2012 (by keep obs
#where val is finite)

njC2[['COUNTY','Dec 2012']]

njC2['Dec 2012']=njC2['Dec 2012']/1000


f, ax = plt.subplots(1, figsize=(6,10))
njC2.plot(column='Dec 2012',legend=True,cmap='RdYlGn_r',scheme='QUANTILES',k=5,
          edgecolor='grey',linewidth=2, ax=ax)
#scheme 'Equal_interval', 'Quantiles', 'Fisher_Jenks'
leg = ax.get_legend()
#leg.set_bbox_to_anchor((1,1,0,0)) #top right
```

```
#leg.set_bbox_to_anchor((0.3,1,0,0)) #top left
#leg.set_bbox_to_anchor((0.3,0.2,0,0)) #bottom left
leg.set_bbox_to_anchor((0.5,0.6,0,0)) #middle left
ax.set_axis_bgcolor("lightslategray") #set bcground col
#ax.set_axis_off() #kills axes
ax.set_xticks([]) #kills xticks
ax.set_yticks([]) #kills xticks
ax.set_title('Watersheds by area ($mi^2$)')
f.savefig('foo.pdf') # can also have png etc extensions


urllib.request.urlretrieve('https://njgin.state.nj.us/oit/gis/download/hsip_colleges.zip','col.zi
p')
import zipfile
zip_ref = zipfile.ZipFile('col.zip', 'r')
zip_ref.extractall()
zip_ref.close()
os.listdir()

col = gpd.read_file('2007_11_30_NJ_COLL_UNIV_njsp.shp')
col.dtypes

f, ax = plt.subplots(1, figsize=(6,10))
# set aspect to equal. This is done automatically
# when using *geopandas* plot on it's own, but not when
# working with pyplot directly.
ax.set_aspect('equal')
njC2.plot(ax=ax, color='white', edgecolor='black',linewidth=1)
#col.plot(ax=ax, marker='o', color='red', markersize=5)
col.plot(column='ENROLL',markersize=50,legend=True,cmap='RdYlGn_r',scheme='QUANTILES',k=5,ax=ax)
ax.set_xticks([]) #kills xticks
ax.set_yticks([]) #kills xticks
ax.set_title('Watersheds by area ($mi^2$)')
f.savefig('foo.pdf') # can also have png etc extensions

#BUG!!the vals shown are way to small, there is like 35k! hopefully fixed in new
#edition of geopandas!!


### subplot with several say pop2010 and pop2015
njCounties['POP2010'].max() # aha so make max like 1m uniform for both
njCounties['POP1980'].min() #and min like 50k
f=plt.figure(figsize=(10,6))
#plt.subplots_adjust(wspace=0)
ax1=plt.subplot(121,title="pop  2010") #1x2grid first subplot
njCounties.plot(column='POP2010',legend=True,cmap='Blues',
          edgecolor='grey',linewidth=2, vmin=50000, vmax=1000000, ax=ax1)

ax2=plt.subplot(122,title="pop  1980",sharex=ax1,sharey=ax1)
njCounties.plot(column='POP1980',legend=True,cmap='Blues',
          edgecolor='grey',linewidth=2,vmin=50000, vmax=1000000, ax=ax2)
ax1.set_xticks([]) #kills xticks
ax1.set_yticks([]) #kills xticks
plt.show()
f.savefig('foo.pdf') # can also have png etc extensions

#NOT SURE WHAT THIS WAS SUPPOSED TO BE
# f, ax = plt.subplots(1,2, figsize=(6,10))
# njC2.plot(column='Dec 2012',legend=True,cmap='RdYlGn_r',scheme='QUANTILES',k=5,
#           edgecolor='grey',linewidth=2, ax=ax)
# #scheme 'Equal_interval', 'Quantiles', 'Fisher_Jenks'
# leg = ax.get_legend()
# #leg.set_bbox_to_anchor((1,1,0,0)) #top right
# #leg.set_bbox_to_anchor((0.3,1,0,0)) #top left
# leg.set_bbox_to_anchor((0.3,0.2,0,0)) #bottom left
# #ax.set_axis_off() #kills axes
# ax.set_xticks([]) #kills xticks
# ax.set_yticks([]) #kills xticks
# ax.set_title('Watersheds by area ($mi^2$)')
# f.savefig('foo.pdf') # can also have png etc extensions
```

```
#### heatmpas http://nbviewer.jupyter.org/gist/perrygeo/c426355e40037c452434

TODO repost *all* data on my website!
import geopandas as gpd
import numpy as np
from scipy import ndimage
import matplotlib.pylab as pylab
import matplotlib.pyplot as plt
pylab.rcParams['figure.figsize'] = 8, 6

urllib.request.urlretrieve('https://docs.google.com/uc?id=1T_n1y_Mj5yQiWpZwrbuuFFwmIVJ2QWFZ&expor
t=download','dirtyNJ.zip')
import zipfile
zip_ref = zipfile.ZipFile('dirtyNJ.zip', 'r')
zip_ref.extractall()
zip_ref.close()
os.listdir()
#New_Jersey_Contaminated_Sites.shp

pts = gpd.GeoDataFrame.from_file('New_Jersey_Contaminated_Sites.shp')
pts.plot()

def heatmap(d, bins=(100,100), smoothing=1.3, cmap='jet'):
    def getx(pt):
        return pt.coords[0][0]

    def gety(pt):
        return pt.coords[0][1]

    x = list(d.geometry.apply(getx))
    y = list(d.geometry.apply(gety))
    heatmap, xedges, yedges = np.histogram2d(y, x, bins=bins)
    extent = [yedges[0], yedges[-1], xedges[-1], xedges[0]]

    logheatmap = np.log(heatmap)
    logheatmap[np.isneginf(logheatmap)] = 0
    logheatmap = ndimage.filters.gaussian_filter(logheatmap, smoothing, mode='nearest')

    plt.imshow(logheatmap, cmap=cmap, extent=extent)
    plt.colorbar()
    plt.gca().invert_yaxis()
    plt.axis('off') #aok invention!
    plt.show()

heatmap(pts, bins=50, smoothing=1.5)

f, ax = plt.subplots()
ax.set_aspect('equal')
njCounties.plot(ax=ax, color='none', edgecolor='white',linewidth=3)
heatmap(pts, bins=50, smoothing=1.5) #MAYBE figure out what that means!
#and legend--eg how many dots per square i guess


### geocoding--skip for now; best use qgis anyway bc can neatly chk with goog openlayers

urllib.request.urlretrieve('https://sites.google.com/site/adamokuliczkozaryn/gis_int/apartments-f
or-rent.xls','apt.xls')
apt=pd.read_excel('apt.xls')
apt.head(1)
apt.dtypes
apt1=apt[['Philadelphia Apartments','Address','City','State','Zip']][0:11]
apt1

import pip
pip.main(['install', 'geopy','--user'])
from geopy import geocoders
g = geocoders.Nominatim()  #GoogleV3()
place = g.geocode("401 cooper st camden nj")
place
place[0]
place[1]
place[1][0]
```

```
place[1][1]

apt1['lat']=0.000000000 #that way its a float :)
apt1['lon']=0.000000000
for i in enumerate(apt1['Philadelphia Apartments']):
    print(i[0],i[1])


#BUG  Nominatim hangs up on 400 S 48th St Philadelphia PA 19143--need err exception handling
for i in enumerate(apt1['Philadelphia Apartments']):
    a=str(apt1['Address'][i[0]])+" "+str(apt1['City'][i[0]])+" "+str(apt1['State'][i[0]])+" "+str
(apt1['Zip'][i[0]])
    print(a)
    place = g.geocode(a)
    apt1['lat'][i[0]]=place[1][0]
    apt1['lon'][i[0]]=place[1][1]


apt1[['lat','lon']]

>>>then i guess save to kml or sth and load back with geopandas

#advice: want to make them nicer, customize, eg put text on them and annotate:
# just use free 'inkskape' to edit saved figure

#### from gisMap.py

import pip
#pip.main(['install', 'folium'])
#pip.main(['install', 'geopy'])
#or:
pip.main(['install', 'folium','--user'])
pip.main(['install', 'geopy','--user'])


import urllib
import webbrowser
import folium as f
from folium.plugins import MarkerCluster, HeatMap

from geopy import geocoders
#g = geocoders.GoogleV3()
g = geocoders.Nominatim() # goog bertter but limited queries per 24hrs



#import geopandas as gp
import pandas as pd

import json


##################
# FOLIUM LEAFLET #
##################

'''
saves maps as html using popular leaflet/openstreetmap engine; can print them
out then to pdf or paper, or just embed html on your website
'''

####examples from https://pypi.python.org/pypi/folium ; new syntax from https://media.readthedocs
.org/pdf/folium/latest/folium.pdf

import webbrowser
webbrowser._tryorder
webbrowser.open('http://www.google.com')
webbrowser.open('m.html'); webbrowser.open('m.html') # can rerun it or simply refersh manually
webbrowser._browsers.items()

# https://docs.python.org/3/library/webbrowser.html#webbrowser.register
webbrowser.get('chrome').open('http://www.google.com')
```

```
webbrowser.get('mozilla').open('http://www.google.com')
webbrowser.get('firefox').open('http://www.google.com')
webbrowser.get('windows-default').open('http://www.google.com')


#m = f.Map(location=[45.5236, -122.6750]) #open street map is default
#m.save('m.html')  #; webbrowser.open('m.html')
#webbrowser.get('firefox').open_new_tab('m.html') #; webbrowser.open('m.html')

place = g.geocode("camden nj") #first geocode location
place[1]

f.Map(location=place[1]).save('m.html'); webbrowser.open('m.html') # and map it; (refresh file in
 firefox)

# zomm more,focus on roads--instad of defaul 'open street map' use 'Stamen Toner'
f.Map(location=place[1], tiles='Stamen Toner', zoom_start=16).save('m.html'); webbrowser.open('m.
html')

# and little brighter
#f.Map(location=place[1], tiles='Stamen Terrain', zoom_start=16).save('m.html'); webbrowser.open(
'm.html')

# lat/lon popup: .lat_lng_popover()
#m=f.Map(location=place[1], zoom_start=16)
#m.add_child(f.LatLngPopup())
#m.save('m.html'); webbrowser.open('m.html')


###adding markup by hand--easy! good for few u/a


m = f.Map(location=g.geocode("401 cooper st camden nj")[1], zoom_start=18)
m.add_child(f.Marker(g.geocode("401 cooper st camden nj 08120")[1], popup='DPPA'))
m.add_child(f.Marker(g.geocode("321 cooper st camden nj 08102" )[1], popup='bbb'))
#m.add_child(f.CircleMarker(g.geocode("waterfront camden nj")[1], popup='waterfront'))
m.save('m.html'); webbrowser.open('m.html')

# '''can pick color, icon
# icons: http://www.w3schools.com/icons/bootstrap_icons_glyphicons.asp
# colors: http://www.w3schools.com/colors/colors_names.asp
# '''
# m = f.Map(location=g.geocode("401 cooper st camden nj")[1], zoom_start=14)
# m.add_child(f.Marker(g.geocode("401 cooper st camden nj")[1],
#                      popup='DPPA1',icon = f.Icon(icon = 'cloud' ,color = 'green')))
# m.add_child(f.CircleMarker(g.geocode("2 Riverside Dr, Camden, NJ 08103")[1], popup='waterfront'
,color='#8A2BE2',fill_color='#F0F8FF'))
# m.save('m.html'); webbrowser.open('m.html')

# #fixed polygon markers:
# m = f.Map(location=g.geocode("401 cooper st camden nj")[1], zoom_start=14)
# m.add_child(f.RegularPolygonMarker(g.geocode("401 cooper st camden nj")[1],popup='DPPA',fill_co
lor='#8A2BE2', number_of_sides=3, radius=25))
# m.save('m.html'); webbrowser.open('m.html')

#lines

coordinates = [
    [42.3581, -71.0636],
    [37.7833, -122.4167]]

m = f.Map(location=[41.9, -97.3], zoom_start=4)
m.add_children(f.PolyLine(locations=coordinates,weight=8,popup='my commute'))
m.save('m.html'); webbrowser.open('m.html')

#or more real airplane-like--yes, they fly like that; remember earth is round
# coordinates = [
#     [42.3581, -71.0636],
#     [42.82995815, -74.78991444],
#     [43.17929819, -78.56603306],
#     [43.40320216, -82.37774519],
#     [43.49975489, -86.20965845],
```

```
#     [41.4338549, −108.74485069],
#     [40.67471747, −112.29609954],
#     [39.8093434, −115.76190821],
#     [38.84352776, −119.13665678],
#     [37.7833, −122.4167]]

# m = f.Map(location=[41.9, −97.3], zoom_start=4)
# m.add_children(f.PolyLine(locations=coordinates,weight=8,popup='my commute'))
# m.save('m.html'); webbrowser.open('m.html')


# MAYBE (apprently fancy kind of ugly though, and not usefull i guess)
'''
Vincent/Vega Markers
https://pypi.python.org/pypi/folium
'''


### many points, clustering, USEFUL! fancy!

#http://blog.dominodatalab.com/creating−interactive−crime−maps−with−folium/

urllib.request.urlretrieve('https://docs.google.com/uc?id=1k4TT6EvWpMYQCjO0dch3XPRj_zzagsqv&expor
t=download', "SFPD_Incidents_−_Current_Year__2015_.csv")

crimedata = pd.read_csv('SFPD_Incidents_−_Current_Year__2015_.csv')

len(crimedata)

crimedata[1:3]

#for speed purposes
MAX_RECORDS = 1000

#create empty map zoomed in on San Francisco
m = f.Map(location=(37.76, −122.45), zoom_start=12)

loc = []
#add a marker for every record in the filtered data, use a clustered view
for each in crimedata[0:MAX_RECORDS].iterrows():
    loc.append([each[1]['Y'],each[1]['X']])


loc[1:3]


m.add_children(MarkerCluster(locations=loc))
m.save('m.html'); webbrowser.open('m.html')
#very cool! hover over it−−pups u in blue a polygon that the aggregate refers to!


#LATERcan also add popups, but guess for speed with so many obs, can ditch it; https://ocefpaf.gi
thub.io/python4oceanographers/blog/2015/12/14/geopandas_folium/



### heatmap http://www.jackboot7.com/visualizing−tweets.html

m = f.Map(location=(37.76, −122.45), zoom_start=12)
m.add_children(HeatMap(loc))
m.save('m.html'); webbrowser.open('m.html')


#a nicer alternative is google heatmap, but req api key and some java editing:
#https://developers.google.com/maps/documentation/javascript/examples/layer−heatmap



################################################################################
#SKIP THE REST; JUST OLD LOSE NOTES: LATER PROCESS THEM, REINCORPORATE ABOVE OR DROP#
#SKIP THE REST; JUST OLD LOSE NOTES: LATER PROCESS THEM, REINCORPORATE ABOVE OR DROP#
```

```
#SKIP THE REST; JUST OLD LOSE NOTES: LATER PROCESS THEM, REINCORPORATE ABOVE OR DROP#
#SKIP THE REST; JUST OLD LOSE NOTES: LATER PROCESS THEM, REINCORPORATE ABOVE OR DROP#
#SKIP THE REST; JUST OLD LOSE NOTES: LATER PROCESS THEM, REINCORPORATE ABOVE OR DROP#
##############################################################################################



#### thematic/choropleth maps


'''
WARNING/MARKETING: only scratchin a surface here; take my GIS class in fall

the idea is that you have some variable and want to show its values with colors
on a map:
GET regular data say in csv and it has to have same variable as in json to merge
GET json geometry/gegraphy/spatial data by googling it, say 'nj counties json' yields:
http://data.ci.newark.nj.us/dataset/new-jersey-counties-polygon/resource/95db8cad-3a8c-41a4-b8b1-
4991990f07f3
and i saved it on my website, and we will pull from there

transfer from one geograpphic format to another is difficult! converting to json
doesnt work well! tends to be very long: say 7m coords; better just google json

conversion that doesnt work well:say shp to json:
import requests, zipfile, io
r = requests.get('http://people.hmdc.harvard.edu/~akozaryn/myweb/bounds_nj_shp.zip')
z = zipfile.ZipFile(io.BytesIO(r.content))
z.namelist()
z.extractall()
njcounties = gp.GeoDataFrame.from_file('nj_counties.shp')
with open('njcounties.json', 'w') as f:
    f.write(njcounties.to_json())

or (doesnt work well either) using say https://ogre.adc4gis.com/

json data for states, counties, congres distr
http://eric.clst.org/Stuff/USGeoJSON

more elaboration:
https://blog.ouseful.info/2015/04/17/creating-interactive-election-maps-using-folium-and-ipython-
notebooks/
https://pypi.python.org/pypi/folium

LATER: add polygon popups to thematic map so that can click it and get info like
with markers earlier
'''


##EX: unem and states

#for unemployment data try fred from api.py


urllib.request.urlretrieve('https://raw.githubusercontent.com/python-visualization/folium/master/
examples/us-states.json', "us-states.json")

req = urllib.request.urlopen('https://raw.githubusercontent.com/python-visualization/folium/maste
r/examples/us-states.json')
encoding = req.headers.get_content_charset()
obj = json.loads(req.read().decode(encoding))

for key in obj:
    print(key)

obj['type']

for key in obj['features'][1]:
    print(key)
```

```
obj['features'][1]['id']
#and open in webbrowser raw data--easy to see!!

m = f.Map(location=g.geocode("kansas")[1], zoom_start=5)
m.choropleth(geo_path='us-states.json', line_color='blue', fill_color='white', line_weight=1,fill
_opacity=0.1)
m.save('m.html'); webbrowser.open('m.html')

urllib.request.urlretrieve('https://raw.githubusercontent.com/python-visualization/folium/master/
examples/US_Unemployment_Oct2012.csv', "un.csv")

un= pd.read_csv('un.csv')
un

#have a look at geog data
geom=pd.read_json('us-states.json')
geom


m = f.Map(location=g.geocode("kansas")[1], zoom_start=5) #center in mid US
m.choropleth(geo_path='us-states.json',
data=un, columns=['State', 'Unemployment'], #1st is id, 2nd mapping
key_on='feature.id', #keys from json
 line_color='blue',
 line_weight=1,fill_opacity=0.5,
#threshold_scale=[2,4,6,8,10],
fill_color='YlOrBr', # ?BuGn?, ?BuPu?, ?GnBu?, ?OrRd?, ?PuBu?, ?PuBuGn?, ?PuRd?, ?RdPu?, ?YlGn?, ?YlGnBu?, ?Y
legend_name = 'Number of') #doesnt wok on my firefox
m.save('m.html'); webbrowser.open('m.html')



##EX: nj counties



urllib.request.urlretrieve('http://people.hmdc.harvard.edu/~akozaryn/myweb/njCounties.geojson.jso
n', "njCounties.geojson.json")

m = f.Map(location=g.geocode("new jersey")[1], zoom_start=7)
m.choropleth(geo_path='njCounties.geojson.json', line_color='blue', fill_color='white', line_weig
ht=1,fill_opacity=0.1)
m.save('m.html'); webbrowser.open('m.html')

# now add some 'regular data', say from http://www.countyhealthrankings.org/rankings/data
urllib.request.urlretrieve('http://www.countyhealthrankings.org/sites/default/files/2014CountyHea
lthRankingsNationalData.xls', "2014CountyHealthRankingsNationalData.xls")

# FIPS as string so leading zeros are kept!
converters = {'FIPS': str} #we'll use count stringl; but just in case
df = pd.read_excel('2014CountyHealthRankingsNationalData.xls', 'Ranked Measure Data', skiprows=1,
 converters=converters)
df.head()
df.dtypes

#select relevant stuff
df['State'].value_counts()
len(df[df.State=='New Jersey'])
dfNJ=df[df.State=='New Jersey']
dfNJ

dfNJalc = dfNJ.copy()[['FIPS', 'County', '# Alcohol-Impaired Driving Deaths', '# Driving Deaths',
 '% Alcohol-Impaired']]
#df_out.dropna(inplace=True)
#df_out['# Alcohol-Impaired Driving Deaths'] = df_out['# Alcohol-Impaired Driving Deaths'].astype
(int, copy=False)
#df_out['# Driving Deaths'] = df_out['# Driving Deaths'].astype(int, copy=False)
dfNJalc


#now have another look at json
#geom=pd.read_json('njCounties.geojson.json')
```

```
#geom.head() #not helpful, let's try another approach

import json
req = urllib.request.urlopen('http://people.hmdc.harvard.edu/~akozaryn/myweb/njCounties.geojson.j
son')
encoding = req.headers.get_content_charset()
obj = json.loads(req.read().decode(encoding))

for i in obj:  #lets loop over to get the elements
    print(i)


obj['type']

obj['features'][1]

for i in obj['features'][1]:
    print(i)

for i in obj['features'][1]['properties']:
    print(i)

obj['features'][1]['properties']['county'] #finally!

dfNJalc = dfNJalc.rename(columns={'County': 'county'}) #name nicely
dfNJalc = dfNJalc.rename(columns={'% Alcohol-Impaired': 'perAlcImp'}) #just ion case get rid of s
pecial chars

##maybe good idea to try merge first and see how it merges

#first simplify
len(obj['features'])
jsonCounties=[]
for i in range(0,len(obj['features'])):
    jsonCounties.append(obj['features'][i]['properties']['county'])

a=sorted(jsonCounties)
b=sorted(dfNJalc['county'])
#set(jsonCounties) & set(dfNJalc['county'])

check=pd.DataFrame(list(zip(a,b)),columns=['json', 'data'])
check['check'] = 0
check['check'][check['json'] == check['data']] = '1'
check

#and print a listing of values to be mapped
dfNJalc[['county','perAlcImp']].sort_values(by='perAlcImp')

## and finally the map
m = f.Map(location=g.geocode("new jersey")[1], zoom_start=8) #center in mid US
m.choropleth(geo_path='njCounties.geojson.json',
data=dfNJalc, columns=['county', 'perAlcImp'], #1st is id, 2nd mapping
key_on='feature.properties.county', #ADJUST THIS!!
 line_color='blue',
 line_weight=1,fill_opacity=0.5,
#threshold_scale=[2,4,6,8,10],
fill_color='YlOrBr', # ?BuGn?, ?BuPu?, ?GnBu?, ?OrRd?, ?PuBu?, ?PuBuGn?, ?PuRd?, ?RdPu?, ?YlGn?, ?YlGnBu?, ?Y
legend_name = 'Number of') #doesnt wok on my firefox
m.save('m.html'); webbrowser.open('m.html')


#maybe better lighter base
m = f.Map(location=g.geocode("new jersey")[1], zoom_start=10, tiles='Stamen Toner')
m.choropleth(geo_path='njCounties.geojson.json',
data=dfNJalc, columns=['county', 'perAlcImp'], #1st is id, 2nd mapping
key_on='feature.properties.county', #ADJUST THIS!!
 line_color='blue',
 line_weight=1,fill_opacity=0.5,
#threshold_scale=[2,4,6,8,10],
fill_color='YlOrBr', # ?BuGn?, ?BuPu?, ?GnBu?, ?OrRd?, ?PuBu?, ?PuBuGn?, ?PuRd?, ?RdPu?, ?YlGn?, ?YlGnBu?, ?Y
legend_name = 'Number of') #doesnt wok on my firefox
m.save('m.html'); webbrowser.open('m.html')
```