

# Lecture 5: Spatial Analysis of Census Data

Today

- 1) Summary Network Science and Models
- 2) Cenpy Library
- 3) Basic Calculations with Census Data
- 4) Guest Speaker (Analyzing Philippine Election Data)

# Summary Network Science and Models

Which properties have we studied?

1) Degree

2) Clustering Coefficient

3) Average Shortest Path

# Random networks: summary

A Random Graph defined by  $(p, N)$ , is built connecting each pair of nodes with probability  $p$ . its analytical properties were shown in Lecture 3. Summarized here:

- Degree distribution Bell shaped curve around average,  $P(k) = e^{-\langle k \rangle} \frac{\langle k \rangle^k}{k!}$
- $\langle k_{\text{random}} \rangle = p(N-1) = 2L/N$ , where  $L$  is the number of links and  $N$  and  $p$  are input parameters of the model
- Average shortest path length  $\langle l_{\text{random}} \rangle \sim \log(N)/\log(k)$
- Clustering Coefficient  $\langle C_{\text{random}} \rangle = p$

Note: You can generate desired  $\langle k \rangle$  by selecting  $p$  and  $N$ ... In the assignment you will use  $\langle k \rangle$  from empirical networks to select  $p$  (see next slide)

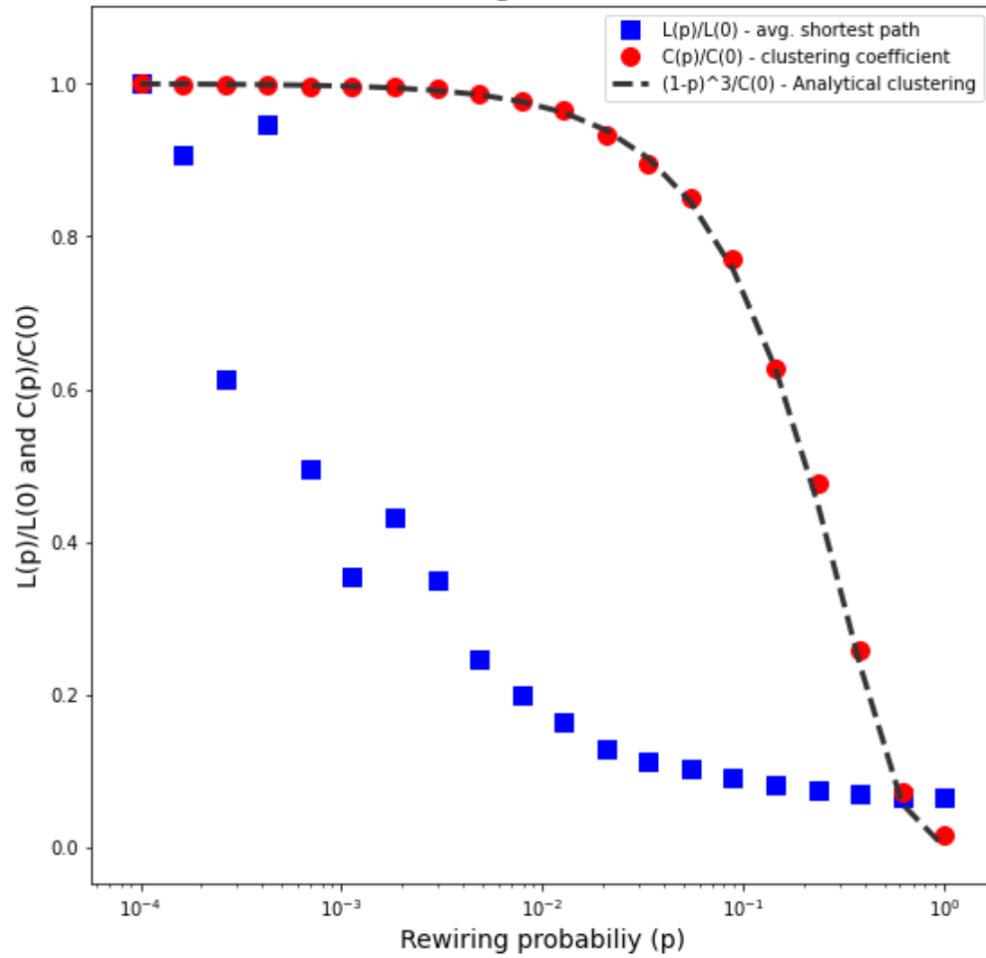
---

# Small World Model: summary

- Degree distribution Delta shaped curve around average
- $\langle k \rangle$  is an input parameter
- Average shortest path length  $\langle l \rangle \sim \log(N)/\log(k)$  for  $p$  in the range  $(0.01, 0.1)$ , e.g. when we rewire a small fraction of the links
- Clustering Coefficient  $C(p,k,n) = C(0,k,n)(1-p)^3$

---

Characteristics of Watts-Strogatz model with  $n=1000$  and  $k=10$



```

for p in p_range:

    # Create a Watts-Strogatz graph
    graph = nx.watts_strogatz_graph( n, k, p )

    # Compute the average shortest-path length L(p)
    sp_length.append( nx.average_shortest_path_length( graph ) )

    # Compute the clustering coefficient C(p)
    clustering.append( nx.average_clustering( graph ) )

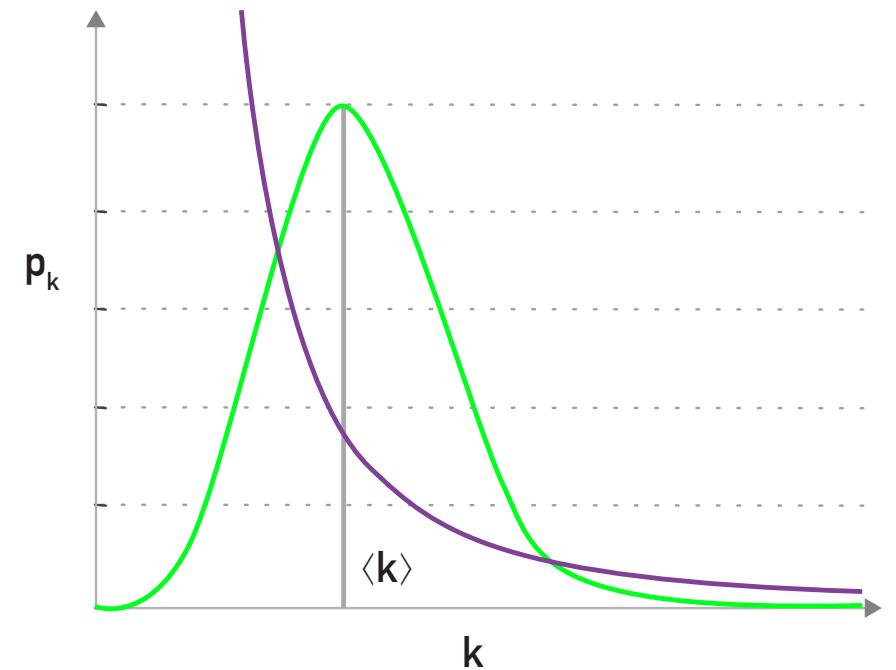
plt.semilogx(p_range[1:] , [L / sp_length[0] for L in sp_length[1:] ] ,
plt.semilogx(p_range[1:] , [C / clustering[0] for C in clustering[1:] ] ,
plt.semilogx(p_range[1:] , [ (1-p)**3.0 for p in p_range[1:] ] , '--', co

```

Note that `clustering[0]` is the ring with the same  $n$  and  $k$

# Barabasi Model: summary

- Input parameters are  $(N, k_{\min})$
- $\langle k \rangle = 2k_{\min}$  (from slide 27 Lecture 4)
- Degree distribution power law  $P(k) = 2k_{\min}k^{-3}$
- Average shortest path length  $\langle l \rangle \sim \log(N)/\log\langle k \rangle$
- Clustering Coefficient  $C \sim \langle k \rangle/N$

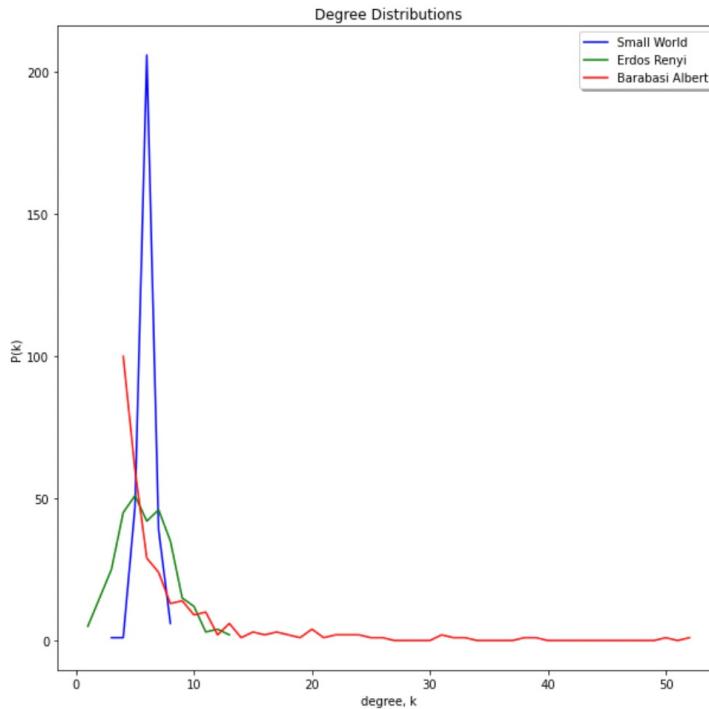


# Use line plot instead of barplots to better compare the histograms

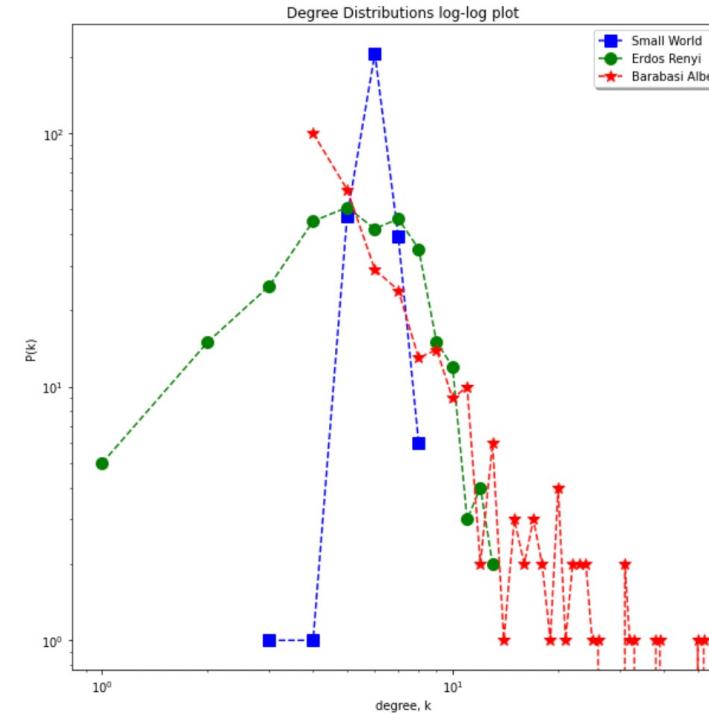
#G1:

```
degs1 = list(dict(nx.degree(gs)).values())
n1, bins1 = np.histogram(degs1, bins = list(range(min(degs1), max(degs1)+1, 1)))
```

plt.plot(bins1[:-1],n1)



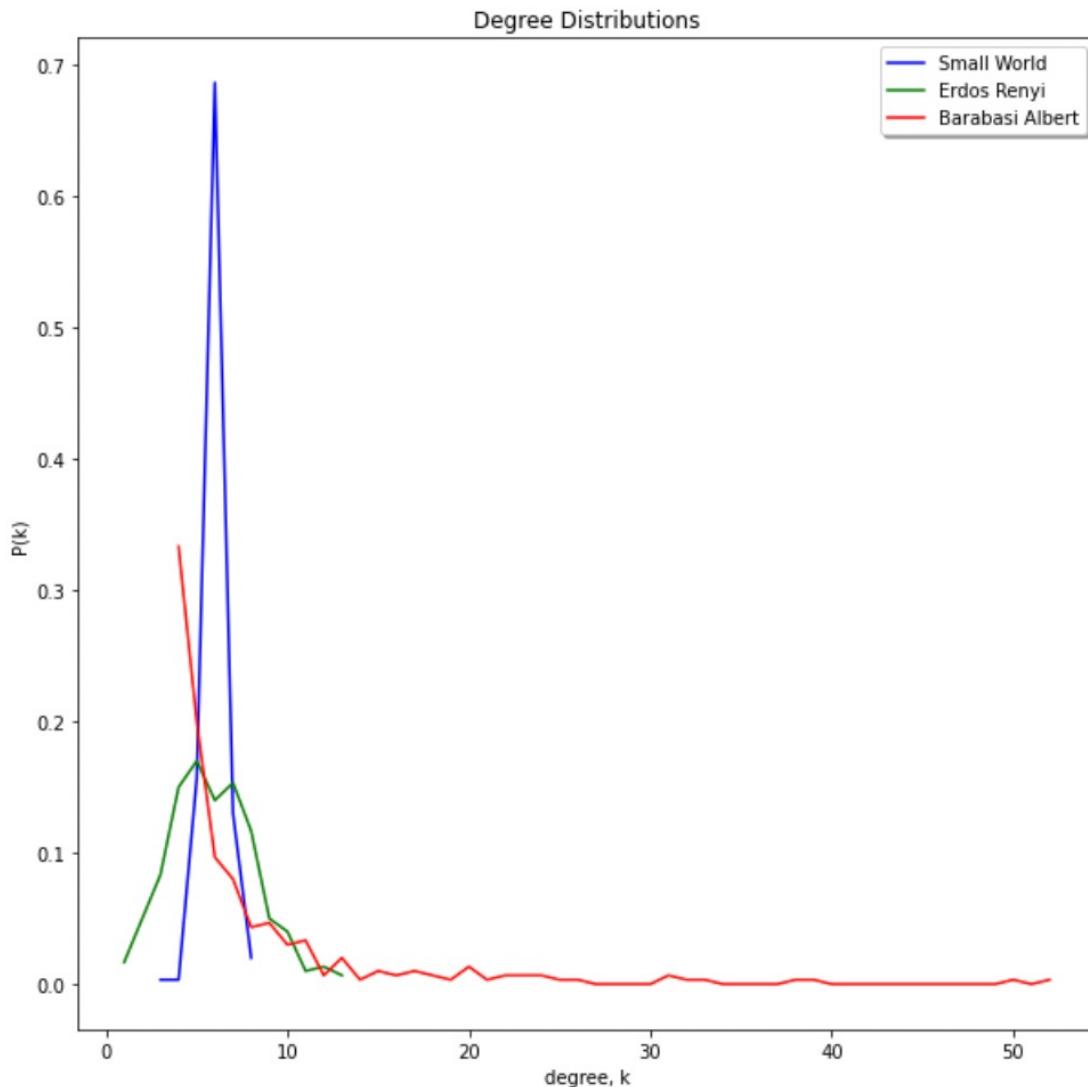
same plot as the left in log-log scale (plt.loglog(bins1[:-1],n1))



gs = nx.watts\_strogatz\_graph( n, k, p ) → select n, k and p to model n,  $\langle k \rangle$  and C from empirical networks  
Ger =nx.erdos\_renyi\_graph(n, p) → select n and p to model n,  $\langle k \rangle$  from empirical networks  
Gba=nx.barabasi\_albert\_graph(n,kmin) → select n and kmin to model n and  $\langle k \rangle$  from empirical networks

# The normalized histograms are the probability density function or distribution of probability

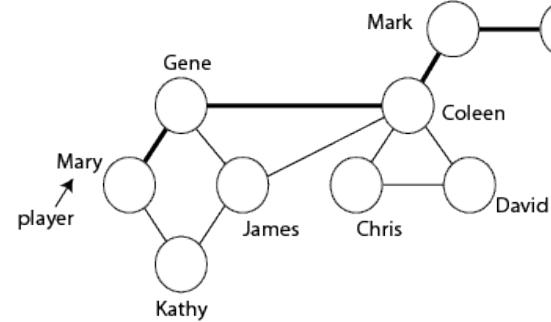
```
1 #G1:  
2 degs1 = list(dict(nx.degree(gs)).values())  
3 n1, bins1 = np.histogram(degs1, bins = list(range(min(degs1), max(degs1)+1, 1))),density='True')  
4  
5 #G2:  
6 degs2 = list(dict(nx.degree(Ger)).values())  
7 n2, bins2 = np.histogram(degs2, bins = list(range(min(degs2), max(degs2)+1, 1))),density='True')  
8  
9 #G3:  
10 degs3 = list(dict(nx.degree(Gba)).values())  
11 n3, bins3 = np.histogram(degs3, bins = list(range(min(degs3), max(degs3)+1, 1))),density='True')  
12  
13 #to plot:  
14 plt.figure(figsize=(10,10)) #use once and set figure size  
15  
16 plt.plot(bins1[:-1],n1,'b-', markersize=10, label="Small World")  
17 plt.plot(bins2[:-1],n2,'g-', markersize=10, label="Erdos Renyi")  
18 plt.plot(bins3[:-1],n3,'r-', markersize=10, label="Barabasi Albert")  
19 plt.legend(loc='upper right', shadow=True)  
20 plt.title('Degree Distributions')  
21 plt.xlabel('degree, k')  
22 plt.ylabel('P(k)')
```



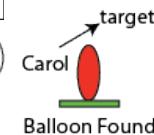
# The Power of Six Degrees

---

# *Power of Social Network: Crowdsourcing*



Carol wins \$2,000  
Mark wins \$1,000  
Coleen wins \$500  
Gene wins \$250

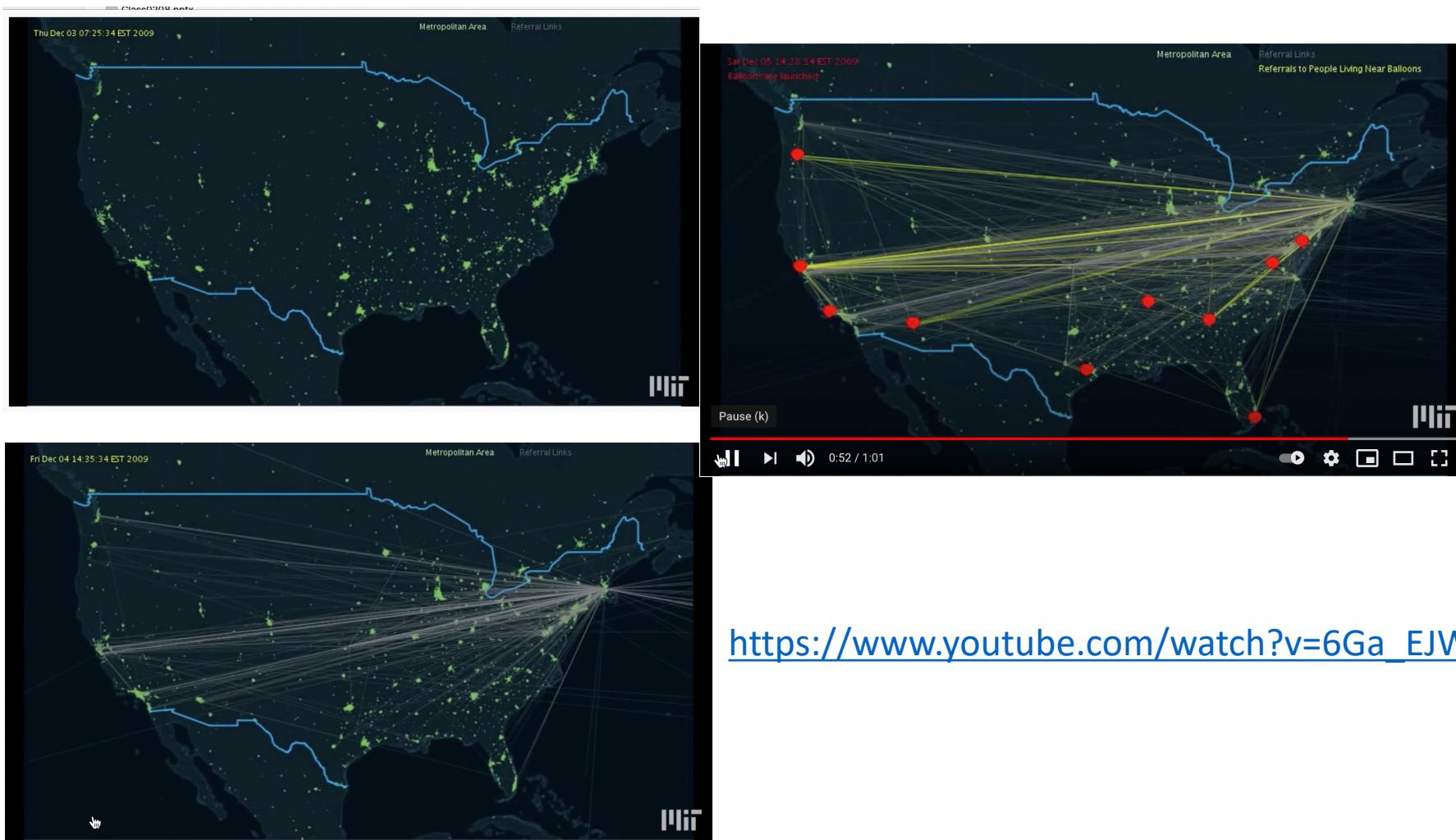


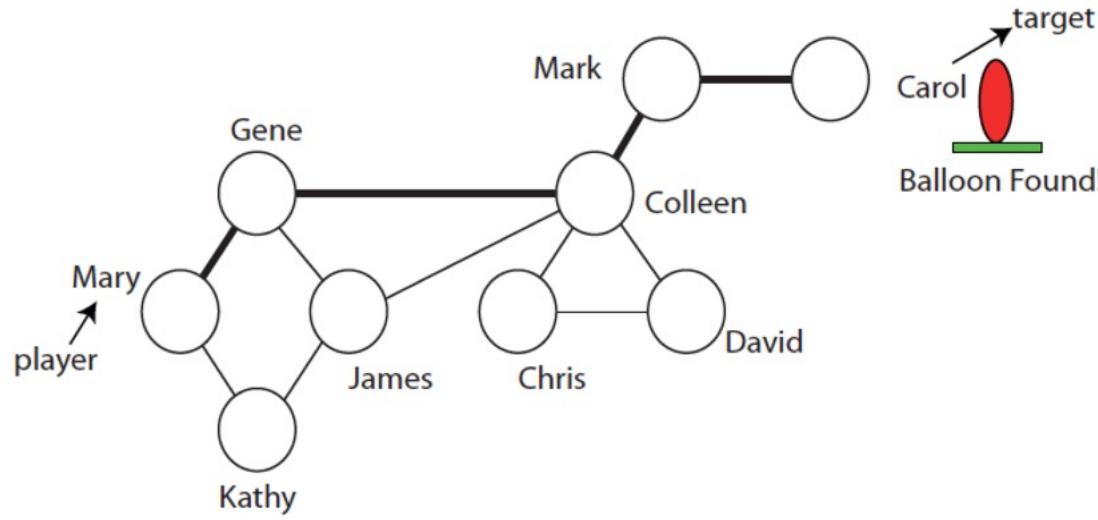
**Red Balloon Experiment: 8 balloons randomly placed in the U.S. were found within 8 hours via social media**



<https://www.cc.com/video/r8x6ag/the-colbert-report-riley-crane>







Consider the sample figure above. If the person who finds the target (Carol) wins \$2,000; and each person in the shortest path to the player wins 1/2 of the money.

What is the total cost of the chain?

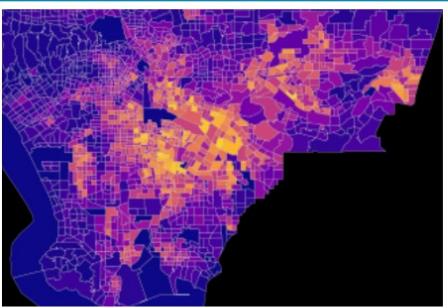
Expected Cost for paths of length 6, and 7 people:  $2000+1000+500+250+125+62.5+31.25= 3968.75$   
 10 balloons would cost ~ 40,000 the prize amount was distributed and they won the game!

# Exploring variables in Census Data

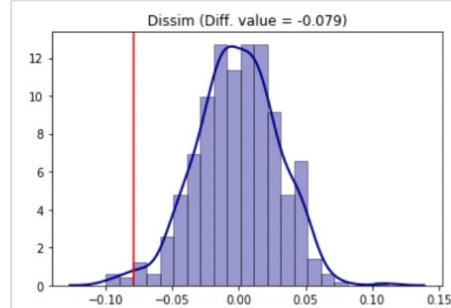
- cenpy\_api.ipynb
- sandiego\_tracts\_calculations.ipynb

# Cenpy

Cenpy (pronounced sen-pie) is a package that automatically discovers US Census Bureau API endpoints and exposes them to Python in a consistent fashion. It also provides easy-to-use access to certain well-used data products, like the American Community Survey (ACS) and 2010 Decennial Census. To get started, check out one of the case studies shown below.

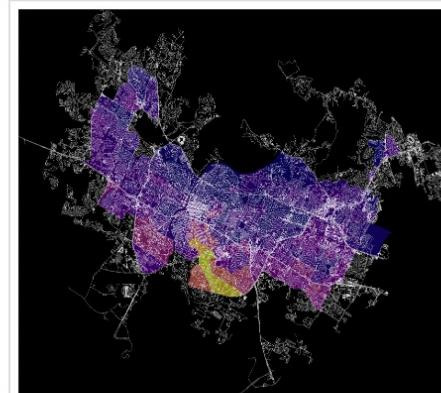


[Getting Data using cenpy](#)



Thus, we see that Austin is significantly less segregated than Phoenix, even when accounting for the uncertainty around estimating the Dissimilarity metric.

Segregation in Time and Space with cenpy and pysal



A Road to Frictionless Urban Data Science: osmnx and cenpy

## The building blocks of cenpy

Cenpy (sen - pie) is a package that exposes APIs from the US Census Bureau and makes it easy to pull down and work with Census data in Pandas.

The Underlying Architecture of cenpy

Cenpy is easiest to install using `conda`, a commonly-used package manager for scientific python. First, [install Anaconda](#).

Then, `cenpy` is available on the `conda-forge` channel. You can install this using the *Anaconda Prompt*, or from within the Anaconda Navigator program. If you want to install the package from within the Anaconda Prompt, you can use:

```
conda install -c conda-forge cenpy
```

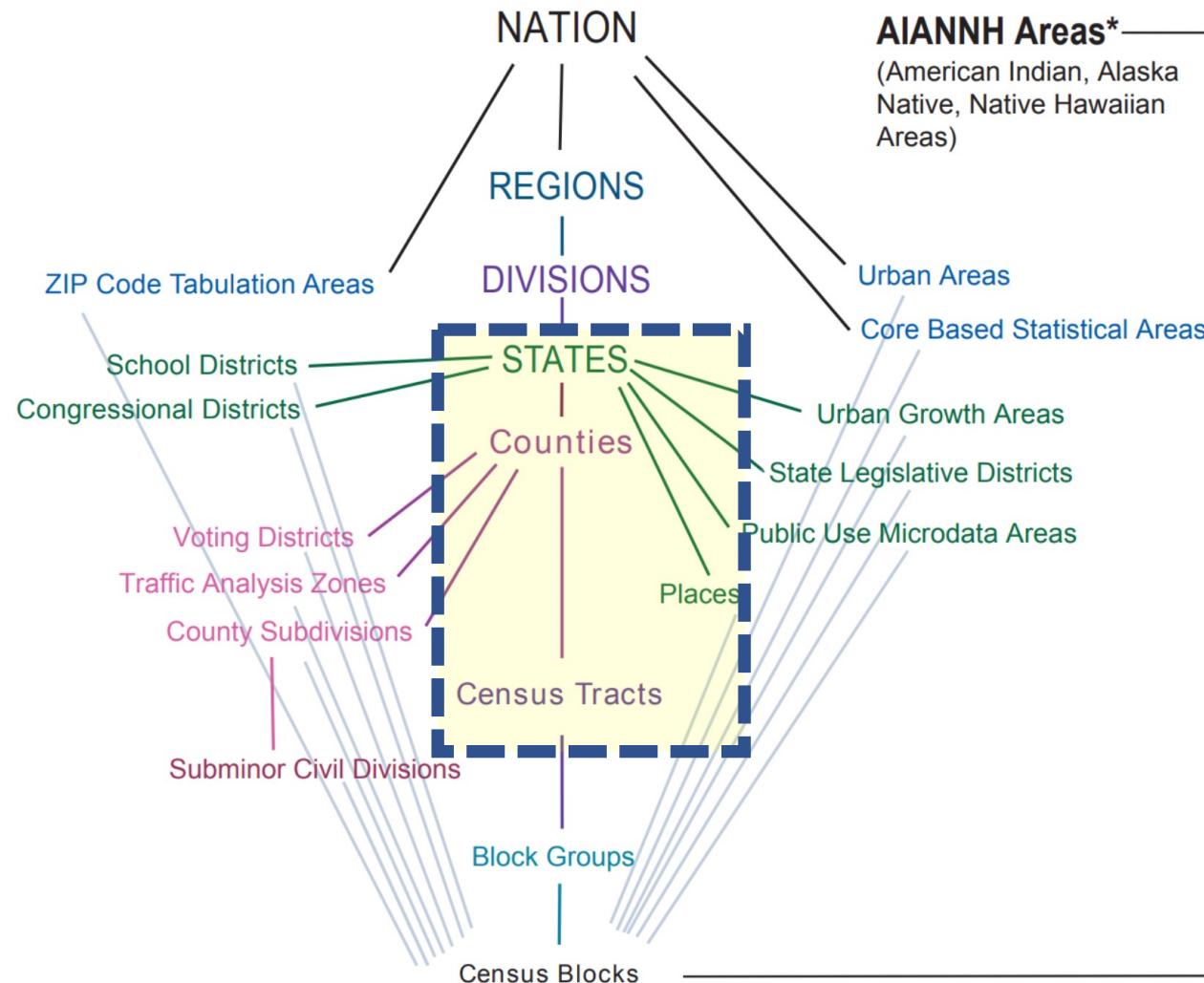
Alternatively, you can install cenpy via `pip`, the python package manager, if you have installed `geopandas` and `rtree`:

```
pip install cenpy
```

<https://cenpy-devs.github.io/cenpy/index.html>

# Geographic Hierarchy of the United States

All of the packages I've seen (including `cenpy` itself) involved a very stilted/specific API query due to the way the census API worked. Basically, it's difficult to construct an efficiently query against the census API without knowing the so-called "geographic hierarchy" in which your query fell:



The main census API does not allow a user to leave middle levels of the hierarchy vague: For you to get a collection of census tracts in a state, you need to query for all the `counties` in that state, then express your query about tracts in terms of a query about all the tracts in those counties. Even `tidycensus` in R requires this

## Cenpy Products

<http://cenpy-devs.github.io/cenpy/api.html>

A product that integrates the data & geographic APIs for the 5-year 2013-2017 ACSs.

---

**cenpy.products.ACS**

The American Community Survey (5-year vintages) from the Census Bureau

---

A product that integrates the data & geographic APIs for the 2010 Census.

---

**cenpy.products.Decennial2010**

The 2010 Decennial Census from the Census Bureau

---

ACS: American Community Survey

ACS5 every 5 years

ACS1 every year

Social
Ancestry
Citizen Voting-Age Population
Citizenship Status
Disability Status
Educational Attainment
Fertility
Grandparents as Caregivers
Language Spoken at Home
Marital History
Marital Status
Migration/Residence 1 Year Ago
Place of Birth
School Enrollment
Undergraduate Field of Degree
Veteran Status; Period of Military Service
Year of Entry

Housing
Bedrooms
Computer and Internet Use
House Heating Fuel
Kitchen Facilities
Occupancy/Vacancy Status
Occupants per Room
Plumbing Facilities
Rent
Rooms
Selected Monthly Owner Costs
Telephone Service Available
Tenure (Owner/Renter)
Units in Structure
Value of Home
Vehicles Available
Year Householder Moved Into Unit
Year Structure Built

Economic
Class of Worker
Commuting (Journey to Work) and Place of Work
Employment Status
Food Stamps/Supplemental Nutrition Assistance Program (SNAP)
Health Insurance Coverage
Income and Earnings
Industry
Occupation
Poverty Status
Work Status Last Year

Demographic
Age; Sex
Group Quarters Population
Hispanic or Latino Origin
Race
Relationship to Householder
Total Population

- **Subject Tables** provide an overview of the estimates available in a particular topic. The data are presented as population counts and percentages. There are over 18,000 variables in this dataset.
- **Data Profiles** contain broad social, economic, housing, and demographic information. The data are presented as population counts and percentages. There are over 1,000 variables in this dataset.

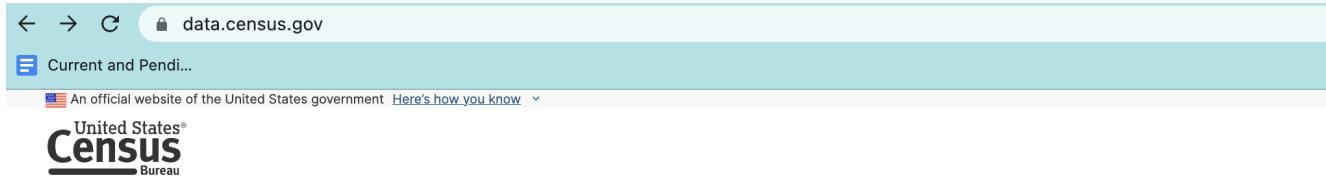
You will select your variables of study following the 3 steps in the next slides

We are limited to the variables available in the ACS or the Decennial2010 products

# Step 1: Explore here variable names:

<https://data.census.gov/>

In this example we search the associated variables for commuting



## Explore Census Data

Learn about America's People, Places, and Economy

A screenshot of the data.census.gov search interface. At the top, there is a search bar with the placeholder 'Find Tables, Maps, and more ...' followed by a microphone icon and a magnifying glass icon. Below the search bar are links for 'Help', 'Feedback', and 'Advanced Search'. A large callout bubble points from the right towards the search bar with the text 'Type a desired search'. Inside the callout bubble, there is a placeholder text 'Try searching for grocery in Georgia in 2017'.

**find  
tables**

# Commuters alameda county

An official website of the United States government [Here's how you know](#)



commuters alameda county



Advanced Search

All

Tables

Maps

Pages

Microdata

Help

FAQ

Fee

1 Filter

Alameda County, California

Clear search

Search for filter

Codes >

Geography >

Surveys >

Topics >

Years >

## Tables

American Community Survey

**S0801** | COMMUTING CHARACTERISTICS BY SEX

View All 23 Products

American Community Survey

**S0802** | MEANS OF TRANSPORTATION TO WORK BY SELECTED CHARACTERISTICS

View All 23 Products

American Community Survey

**S0804** | MEANS OF TRANSPORTATION TO WORK BY SELECTED CHARACTERISTICS FOR WORKPLACE GEOGRAPHY

View All 23 Products

American Community Survey

**S0101** | AGE AND SEX

View All 23 Products

American Community Survey

**S0102** | POPULATION 60 YEARS AND OVER IN THE UNITED STATES

View All 23 Products

American Community Survey

**S0103** | POPULATION 65 YEARS AND OVER IN THE UNITED STATES

View All 23 Products

American Community Survey

**S0501** | SELECTED CHARACTERISTICS OF THE NATIVE AND FOREIGN-BORN POPULATIONS

View All 23 Products

American Community Survey

**S0502** | SELECTED CHARACTERISTICS OF THE FOREIGN-BORN POPULATION BY PERIOD OF ENTRY INTO THE UNITED STATES

View All 23 Products

3,292 Tables, 3,292 Maps, 21 Pages

View: 10 | 25 | 50



County

**Alameda County, California**

**Total Population:** 1,682,353

**Median Household Income:** \$109,729

**Bachelor's Degree or Higher:** 50.1%

**Employment Rate:** 61.9%

**Total Housing Units:** 621,958

**Without Health Care Coverage:** 4.2%

**Total Employer Establishments:** 41,258

**Total Households:** 589,180

**Hispanic or Latino (of any race):** 393,749

[View Profile](#)

## Related Searches

Alameda County, California Business and Economy

Alameda County, California Education

Alameda County, California Employment

Alameda County, California Families and Living Arrangements

Alameda County, California Government

Alameda County, California Health

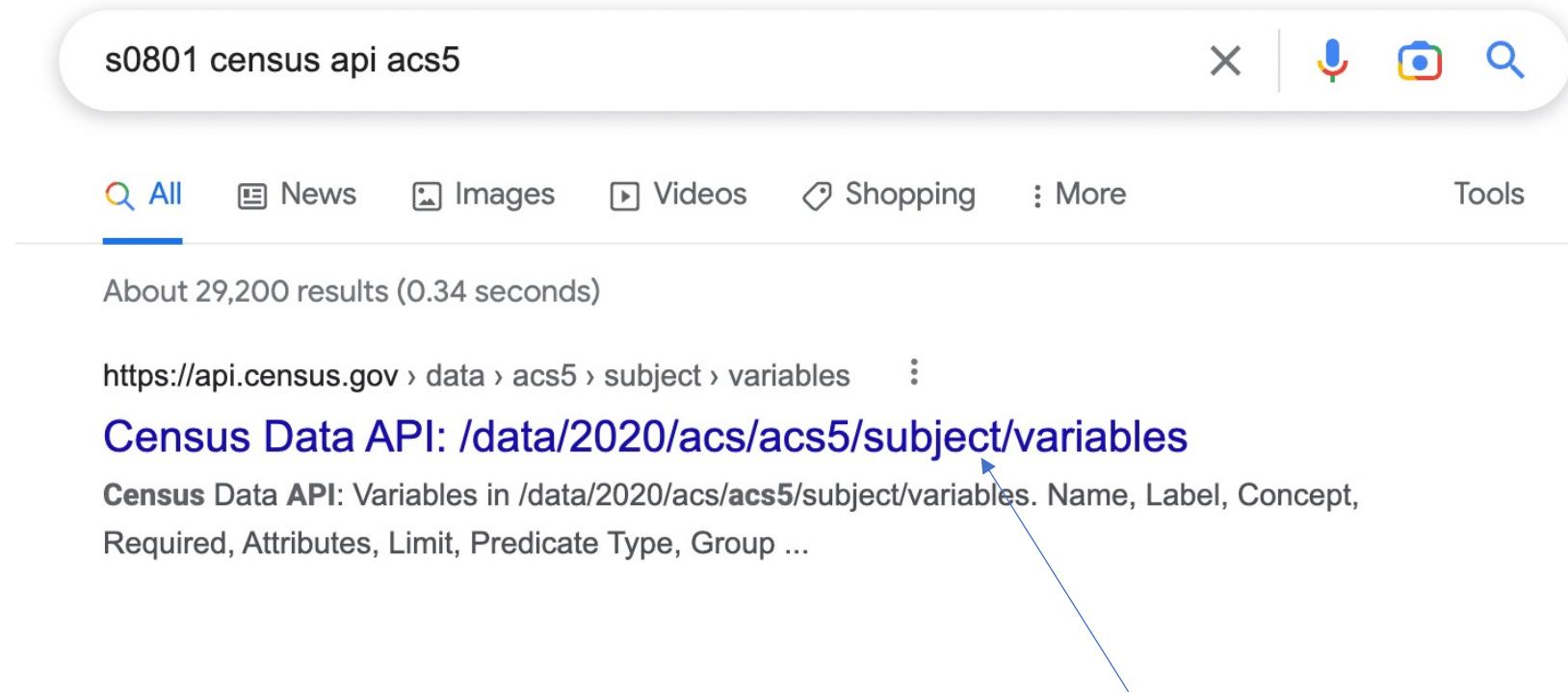
Alameda County, California Housing

Alameda County, California Income and Poverty

Alameda County, California Populations and People

Alameda County, California Race and Ethnicity

## Step 2: google search the variable group name in the census api



A screenshot of a Google search results page. The search bar at the top contains the query "s0801 census api acs5". Below the search bar are navigation links for "All", "News", "Images", "Videos", "Shopping", "More", and "Tools". A status message indicates "About 29,200 results (0.34 seconds)". The first result is a link to the Census Data API documentation: "Census Data API: /data/2020/acs/acs5/subject/variables". A blue arrow points from the text "s0801 is the name of the variable group" down to this link. The snippet below the link describes the endpoint: "Census Data API: Variables in /data/2020/acs/acs5/subject/variables. Name, Label, Concept, Required, Attributes, Limit, Predicate Type, Group ...".

s0801 is the name of the variable group

# Step 3: See the specific variable name to use in cenpy

Current and Pending		
S0702_C05_010E	Estimate!!West!!Region of Current Residence!!PERCENT ALLOCATED!!Residence 1 year ago	MOVERS BETWEEN REGIONS
S0801_C01_001E	Estimate!!Total!!Workers 16 years and over	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_002E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_003E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Drove alone	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_004E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Carpooled	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_005E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Carpooled!!In 2-person carpool	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_006E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Carpooled!!In 3-person carpool	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_007E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Carpooled!!In 4-or-more person carpool	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_008E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Car, truck, or van!!Workers per car, truck, or van	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_009E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Public transportation (excluding taxicab)	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_010E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Walked	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_011E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Bicycle	COMMUTING CHARACTERISTICS BY SEX
S0801_C01_012E	Estimate!!Total!!Workers 16 years and over!!MEANS OF TRANSPORTATION TO WORK!!Taxicab, motorcycle, or other means	COMMUTING CHARACTERISTICS BY SEX

<https://api.census.gov/data/2019/acs/acs5/variables.html>

# cenpy-api.ipynb

- Tutorial to use the Cenpy library
- Cenpy is a Python library that provides a simple interface to access data from the United States Census Bureau's APIs. It enables users to easily download and manipulate US Census data within a Pandas DataFrame.

Median household income: B19013\_001E

Median Contract Rent: B25058\_001E

Households with food stamp: B19058\_002E [1](#)

Households food stamp Estimate Total: B19058\_001E

```
1 from cenpy import products
2 import matplotlib.pyplot as plt
3 import matplotlib as mpl
4 %matplotlib inline
```

```
1 #from_csa means it will extract the map of a Combined Statistical Area.
2 #List of 172 US CSA: https://en.wikipedia.org/wiki/Combined_statistical_area
3 #'B19013_001E' Estimate!!Median household income in the past 12 months (in 2019 inflation-adjusted dollars)
4
5 sj_csa= products.ACS(2019).from_csa('San Jose-San Francisco-Oakland',level='tract',
6 variables=['B19013_001E'])
```

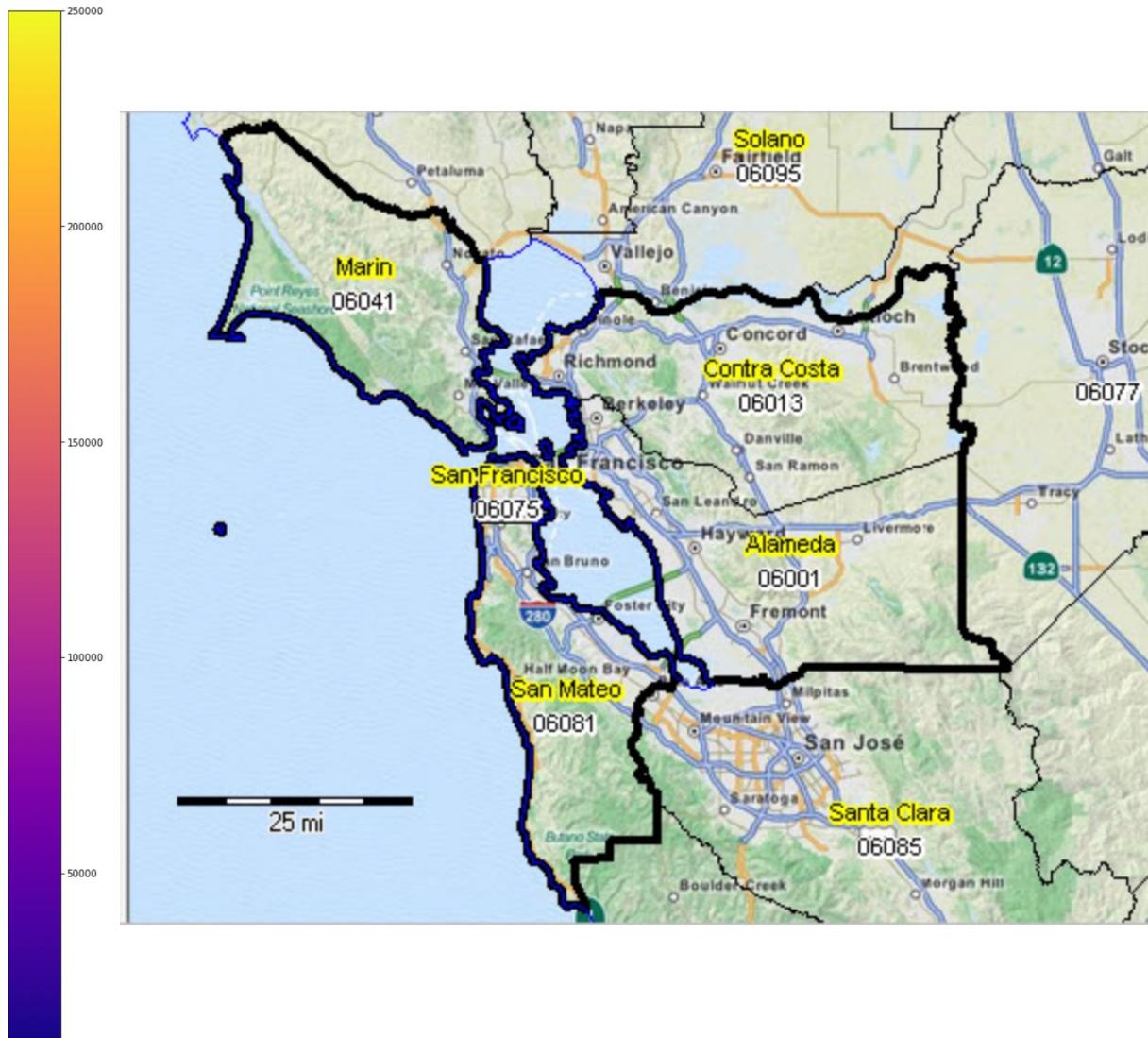
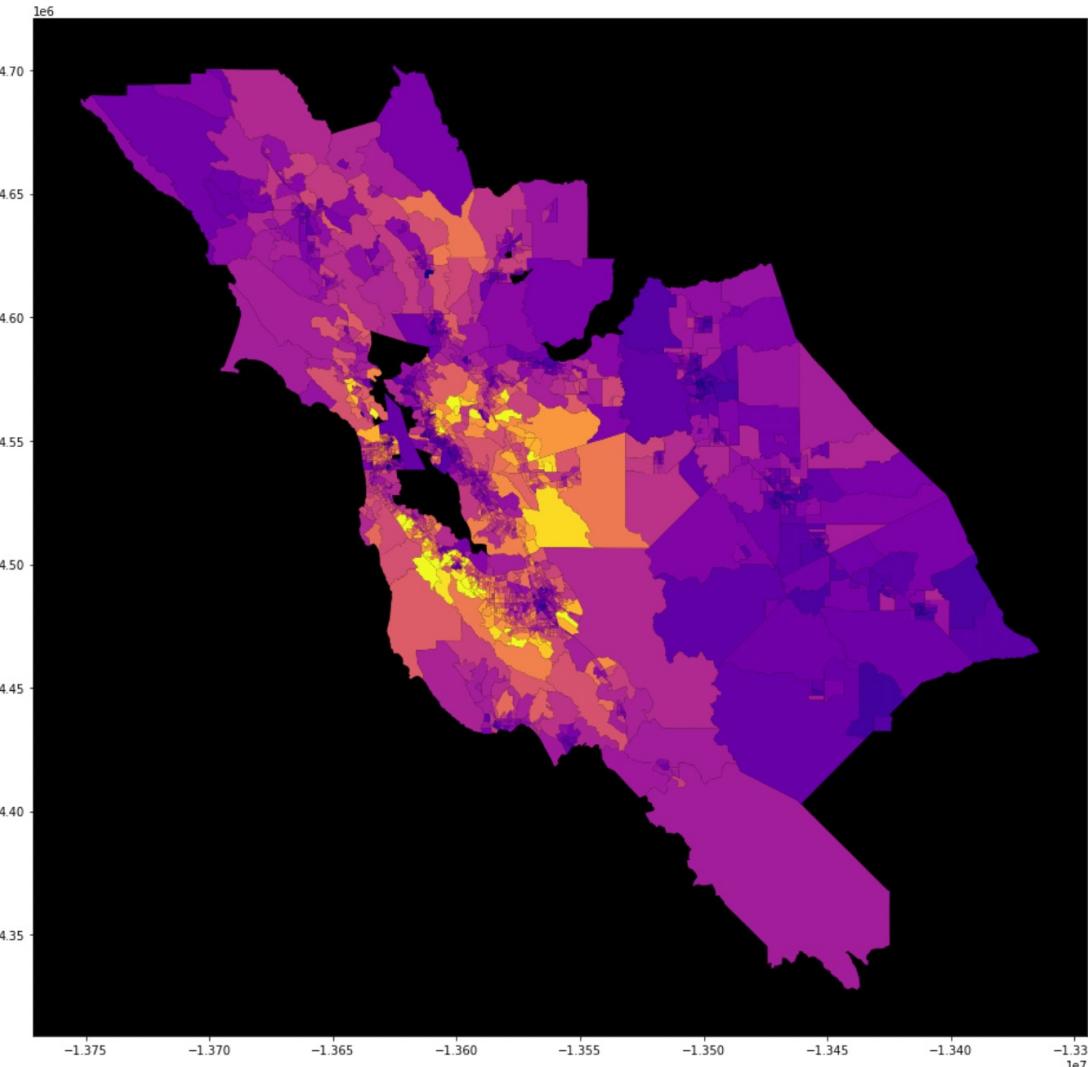
```
1 sj_csa
```

	GEOID	geometry	B19013_001E	NAME	state	county	tract
0	06097990100	POLYGON ((-13762706.610 4687083.180, -13761997...	NaN	Census Tract 9901, Sonoma County, California	06	097	990100
1	06075980401	POLYGON ((-13711647.480 4547791.830, -13711635...	NaN	Census Tract 9804.01, San Francisco County, Ca...	06	075	980401
2	06081612700	POLYGON ((-13603249.680 4500745.130, -13603146...	250001.0	Census Tract 6127, San Mateo County, California	06	081	612700
3	06081612000	POLYGON ((-13598390.690 4504780.430, -13598369...	71234.0	Census Tract 6120, San Mateo County, California	06	081	612000
4	06081612100	POLYGON ((-13598117.850 4502972.090, -13598103...	57627.0	Census Tract 6121, San Mateo County, California	06	081	612100
...	...	...	...	...	...	...	...
1929	06099002701	POLYGON ((-13465889.440 4526163.430, -13465889...	74063.0	Census Tract 27.01, Stanislaus County, California	06	099	002701
1930	06099002605	POLYGON ((-13463858.860 4524354.770, -13463858...	48892.0	Census Tract 26.05, Stanislaus County, California	06	099	002605
1931	06077004902	POLYGON ((-13486050.400 4566015.310, -13485973...	63565.0	Census Tract 49.02, San Joaquin County, Califo...	06	077	004902
1932	06099003909	POLYGON ((-13456035.220 4512110.920, -13456017...	52671.0	Census Tract 39.09, Stanislaus County, California	06	099	003909
1933	06099003905	POLYGON ((-13451642.770 4509065.350, -13451614...	92500.0	Census Tract 39.05, Stanislaus County, California	06	099	003905

In California we have five [combined statistical areas](#), 26 [metropolitan statistical areas](#)

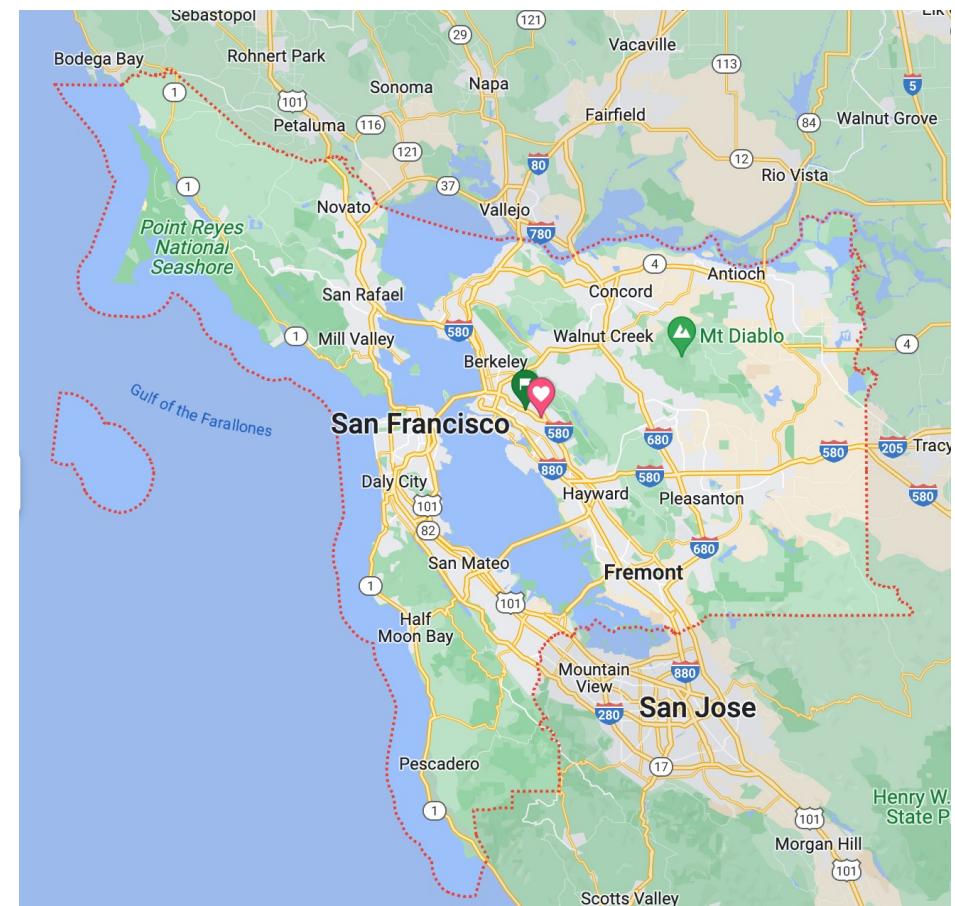
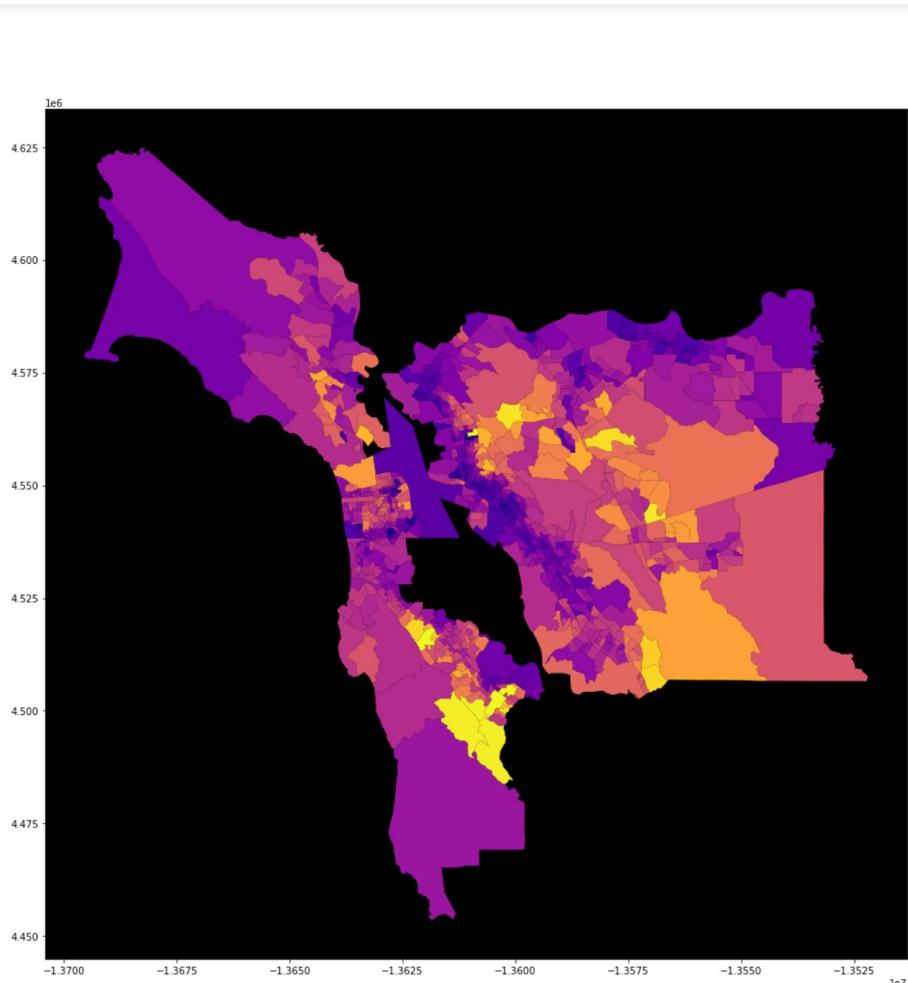
- Cenpy generates a geopandas with rows having geoid of the unit of analysis (county or tract) a polygon, name, and the variable of census extracted
- We can extract several variables comma separated in the variables list

```
1 ## Note drop removes any rows with missing values in the column named 'B19013_001E'.
2 ## sj_csa is a geopandas a table with geoid and geometry that can be displayed as a map
3 f, ax = plt.subplots(1,1,figsize=(20,20))
4 sj_csa.dropna(subset=['B19013_001E'], axis=0).plot('B19013_001E', ax=ax, cmap='plasma',legend=True)
5 ax.set_facecolor('k')
```

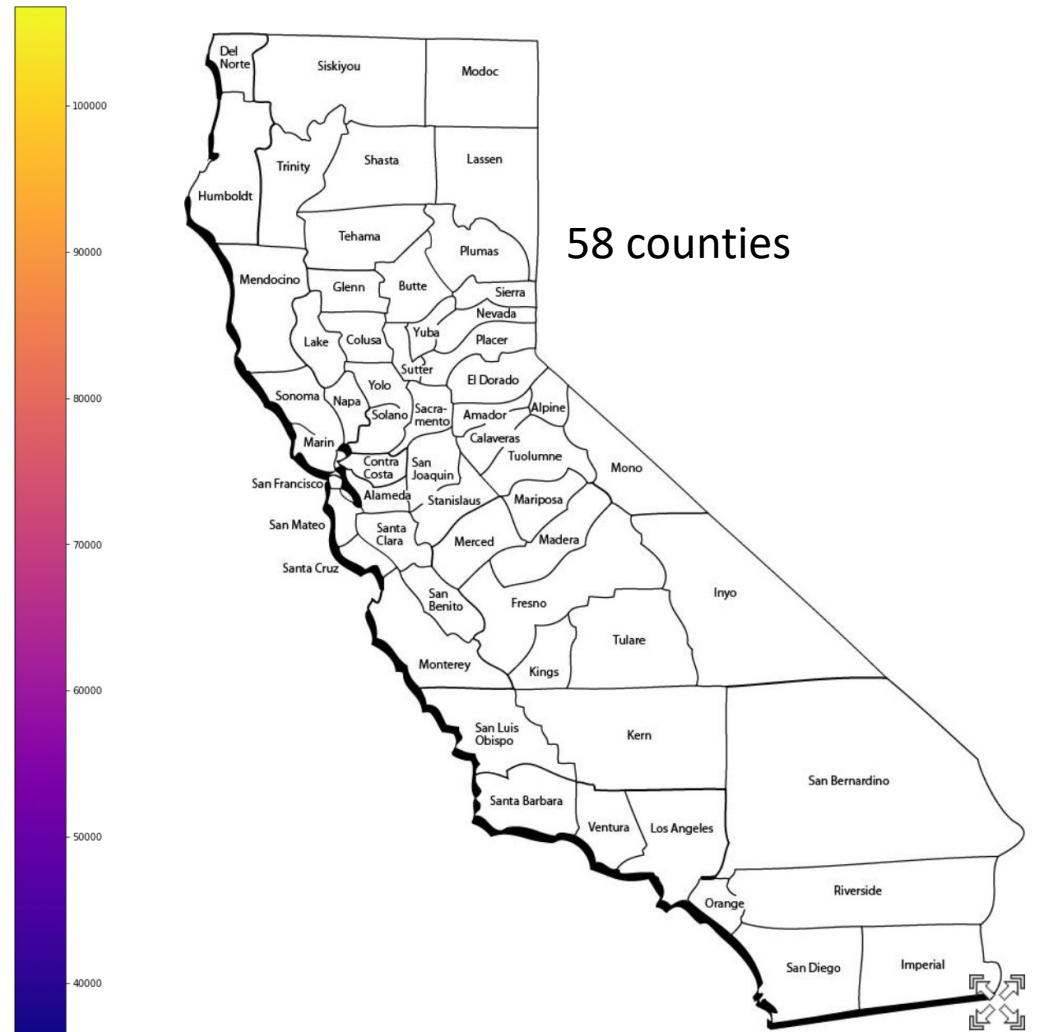
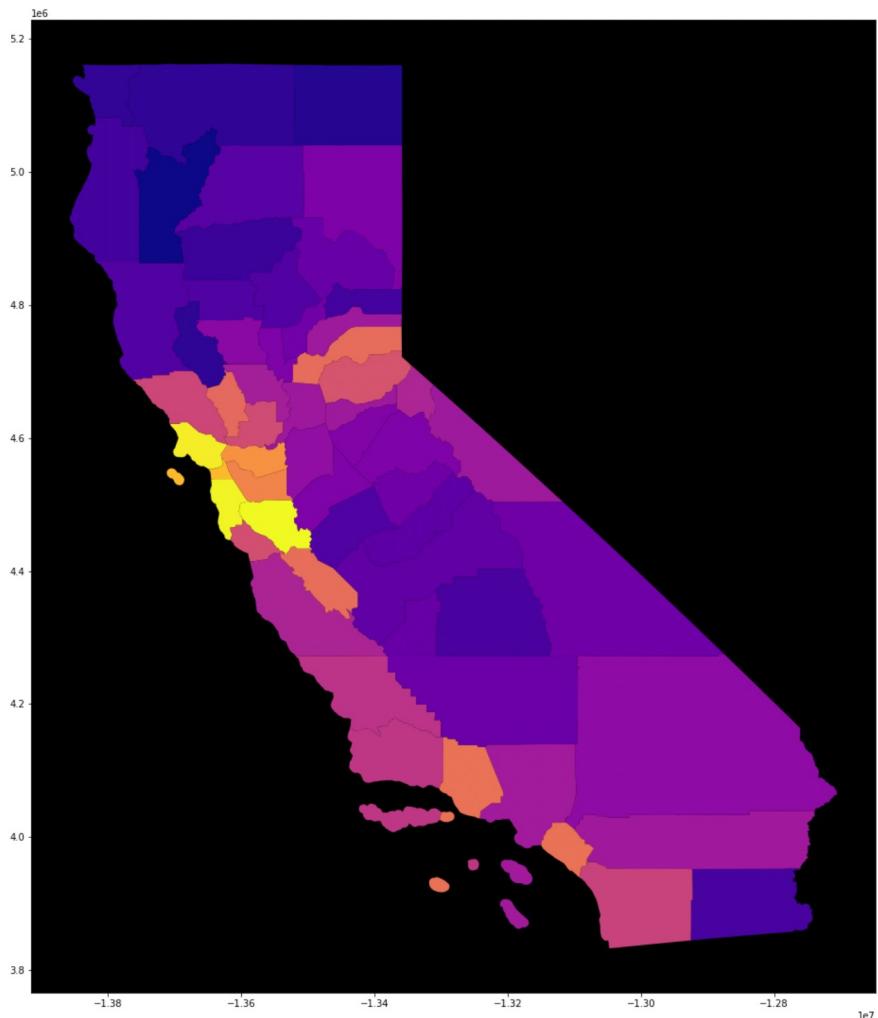


```
1 ###from_msa means it will extract the map of a Metropolitan Statistical Area.
2 ## There are 384 metropolitan statistical areas: https://en.wikipedia.org/wiki/Metropolitan\_statistical\_area
3 sf_msa = products.ACS(2019).from_msa('San Francisco-Oakland-Berkeley',level='tract',
4                                         variables=['B19013_001E'])
```

```
1 f, ax = plt.subplots(1,1,figsize=(20,20))
2 sf_msa.dropna(subset=['B19013_001E'], axis=0).plot('B19013_001E', ax=ax, cmap='plasma',legend=True)
3 ax.set_facecolor('k')
```

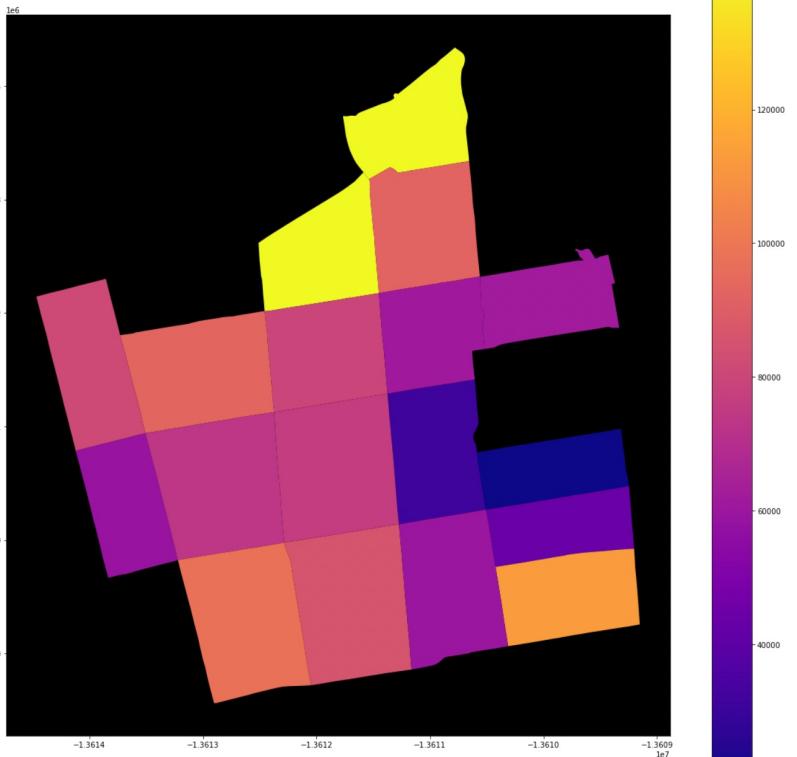


```
1 ca = products.ACS(2019).from_state('California', level='county',
2 variables=['B19013_001E'])
3
4 f, ax = plt.subplots(1,1,figsize=(20,20))
5 ca.dropna(subset=['B19013_001E'], axis=0).plot('B19013_001E', ax=ax, cmap='plasma', legend=True)
6 ax.set_facecolor('k')
```

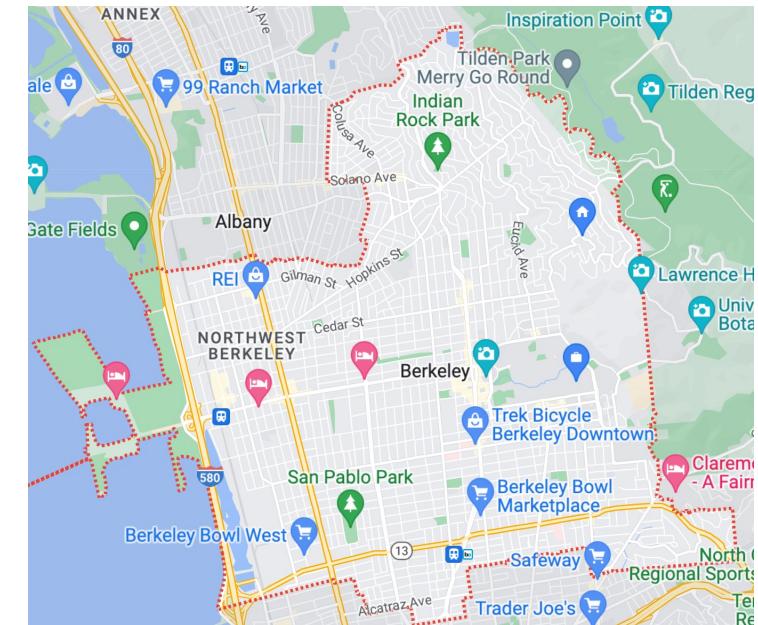


58 counties

## Median Household Income



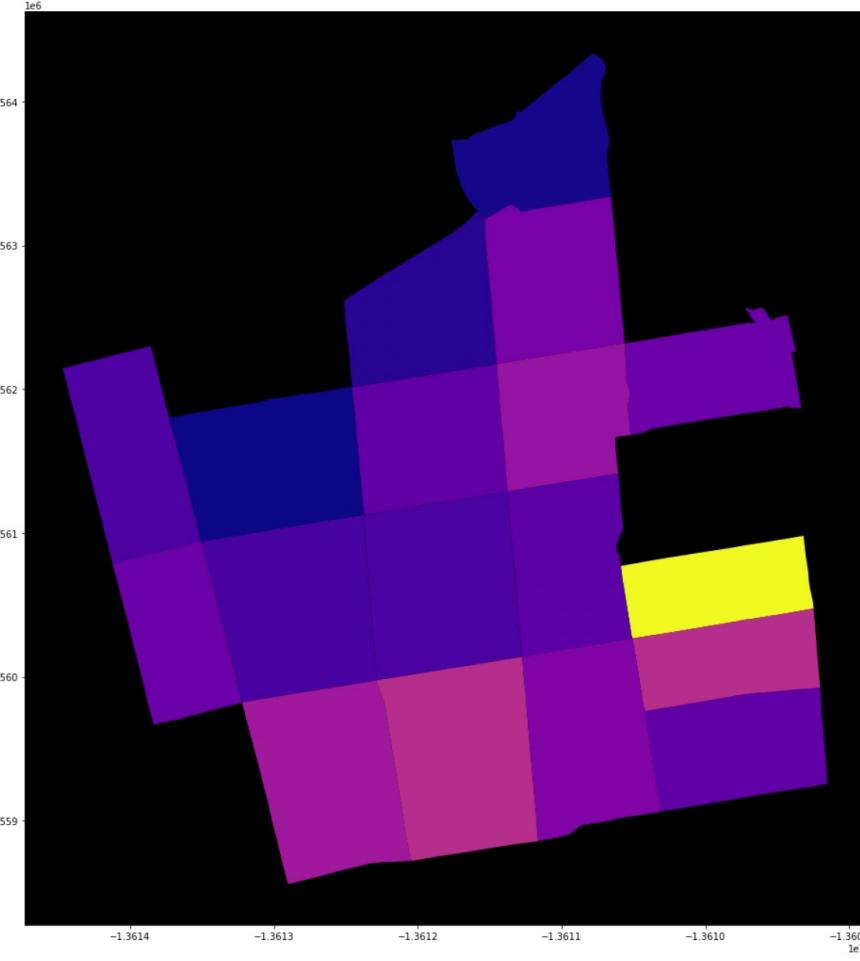
## Median contract rent



### Households food stamp Estimate Total



### Households with food stamp: B19058\_002E





# Available APIs

We plan on adding more of our publicly available datasets. Here you'll find which of our many data sets are currently available via API. To make specific requests for the release of datasets, please sign up and submit your requests on our [Developer Forum](#).

NEW: We now have a machine-readable dataset discovery service available in beta release. Visit our [Discovery Tool page](#) to learn more

[EXPAND ALL](#) | [COLLAPSE ALL](#)

▼ **American Community Survey (ACS)**

▼ **Decennial Census**

▼ **Economic Census**

▼ **Population Estimates and Projections**

▼ **Health Insurance Statistics**

<https://www.census.gov/data/developers/data-sets.html>

CENPY uses the Census API, we can work with their prepared products (ACS and Decennial2010)

Via the US Census API one can get more products

I have code that imports data from the API without Cenpy(ask me if you would like to explore it)

sandiego\_tracts\_calculations.ipynb

We select a set of Census variables that are meaningful for a Data Science Story, gives short friendly names

```
1 import contextily
2 import geopandas
3 import cenpy
4
5 acs = cenpy.products.ACS(2017)
```

## Download Data

- Set variables to download

```
1 vars_to_download = {
2     "B25077_001E": "median_house_value",      # Median house value
3     "B02001_002E": "total_pop_white",        # Total white population
4     "B01003_001E": "total_pop",               # Total population
5     "B25003_003E": "total_rented",            # Total rented occupied
6     "B25001_001E": "total_housing_units",    # Total housing units
7     "B09019_006E": "hh_female",              # Female households
8     "B09019_001E": "hh_total",                # Total households
9     "B15003_002E": "total_bachelor",         # Total w/ Bachelor degree
10    "B25018_001E": "median_no_rooms",        # Median number of rooms
11    "B19083_001E": "income_gini",             # Gini index of income inequality
12    "B01002_001E": "median_age",              # Median age
13    "B08303_001E": "tt_work",                 # Aggregate travel time to work
14    "B19013_001E": "median hh income"        # Median household income
15 }
16 vars_to_download_1 = list(vars_to_download.keys())
```

Source: [https://geographicdata.science/book/data/sandiego/sandiego\\_tracts\\_cleaning.html](https://geographicdata.science/book/data/sandiego/sandiego_tracts_cleaning.html)

Book: **Geographic Data Science with Python**

- Download geometries and attributes

```
: 1 %%time
 2 db = acs.from_msa("San Diego, CA",
 3                     level="tract",
 4                     variables=vars_to_download_1
 5 )
```

```
CPU times: user 1.13 s, sys: 80.1 ms, total: 1.21 s
Wall time: 20.3 s
```

## Metadata

We create a DataFrame that contains information about the variables to be downloaded, including the variable name (stored as "var\_id"), a short name (stored as "short\_name"), the label of the variable, and its associated concept. This information allows us to keep track of what each variable represents.

```
: 1 var_names = acs.variables\  
: 2     .reindex(vars_to_download)\\  
: 3     [[ "label", "concept" ]]\\  
: 4     .reset_index()\\  
: 5     .rename(columns={"index": "var_id"})  
: 6 var_names[ "short_name" ] = var_names[ "var_id" ].map(vars_to_download)
```

```
: 1 var_names
```

	var_id	label	concept	short_name
0	B25077_001E	Estimate!!Median value (dollars)	MEDIAN VALUE (DOLLARS)	median_house_value
1	B02001_002E	Estimate!!Total!!White alone	RACE	total_pop_white
2	B01003_001E	Estimate!!Total	TOTAL POPULATION	total_pop
3	B25003_003E	Estimate!!Total!!Renter occupied	TENURE	total_rented
4	B25001_001E	Estimate!!Total	HOUSING UNITS	total_housing_units
5	B09019_006E	Estimate!!Total!!In households!!In family hous...	HOUSEHOLD TYPE (INCLUDING LIVING ALONE) BY REL...	hh_female
6	B09019_001E	Estimate!!Total	HOUSEHOLD TYPE (INCLUDING LIVING ALONE) BY REL...	hh_total
7	B15003_002E	Estimate!!Total!!No schooling completed	EDUCATIONAL ATTAINMENT FOR THE POPULATION 25 Y...	total_bachelor
8	B25018_001E	Estimate!!Median number of rooms	MEDIAN NUMBER OF ROOMS	median_no_rooms
9	B19083_001E	Estimate!!Gini Index	GINI INDEX OF INCOME INEQUALITY	income_gini
10	B01002_001E	Estimate!!Median age --!!Total	MEDIAN AGE BY SEX	median_age
11	B08303_001E	Estimate!!Total	TRAVEL TIME TO WORK	tt_work
12	B19013_001E	Estimate!!Median household income in the past ...	MEDIAN HOUSEHOLD INCOME IN THE PAST 12 MONTHS ...	median_hh_income

## Process data

While the ACS comes with a large number of attributes, we are not limited to the original variables at hand; we can construct additional variables. This is particularly useful when we want to compare areas that are not very similar in some structural characteristic, such as area or population. For example, a quick look into the variable names shows most variables are counts. For tracts of different sizes, these variables will mainly reflect their overall population, rather than provide direct information about the variables itself. To get around this, we will cast many of these count variables to rates, and use them in addition to a subset of the original variables.

- Replace missing values with columns mean

```
In [33]: 1 filler = lambda col: col.fillna(col.mean())
2 db.loc[:, vars_to_download] = db.loc[:, vars_to_download]\ 
3 .apply(filler)
```

- Replace variable codes with short names

```
In [34]: 1 db = db.rename(columns=vars_to_download)
```

- Calculate area in Sq.Km (we use the [Conus Albers](#) CRS)

```
In [35]: 1 db["area_sqm"] = db.to_crs(epsg=5070).area / 1e6
```

EPSG stands for European Petroleum Survey Group, which is a defunct organization that maintained a database of coordinate reference systems (CRSs)

- Percentage of renter occupied units

```
In [36]: 1 db[ "pct_rented" ] = db[ "total_rented" ] / \
2             (db[ "total_housing_units" ] + \
3             (db[ "total_housing_units" ]==0) * 1
4             )
```

- Percentage of female households

```
In [37]: 1 db[ "pct_hh_female" ] = db[ "hh_female" ] / \
2             (db[ "hh_total" ] + \
3             (db[ "hh_total" ]==0) * 1
4             )
```

Note: if the denominator is 0, it divides by 1

- Percentage without completing Schooling

```
: 1 db[ "pct_bachelor" ] = db[ "total_bachelor" ] / \
 2           (db[ "total_pop" ] + \
 3           (db[ "total_pop" ]==0) * 1
 4           )
```

- Percentage of white population

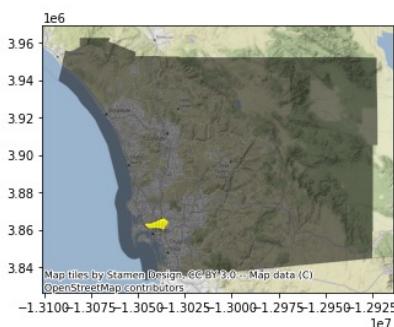
```
: 1 db[ "pct_white" ] = db[ "total_pop_white" ] / \
 2           (db[ "total_pop" ] + \
 3           (db[ "total_pop" ]==0) * 1
 4           )
```

This shows how to sample a selected number of 30 tracts and color them in yellow on top of all the other tracts (628) colored in black with transparency (alpha=0.5)

- Generate indicator for subset of contiguous 30 tracts

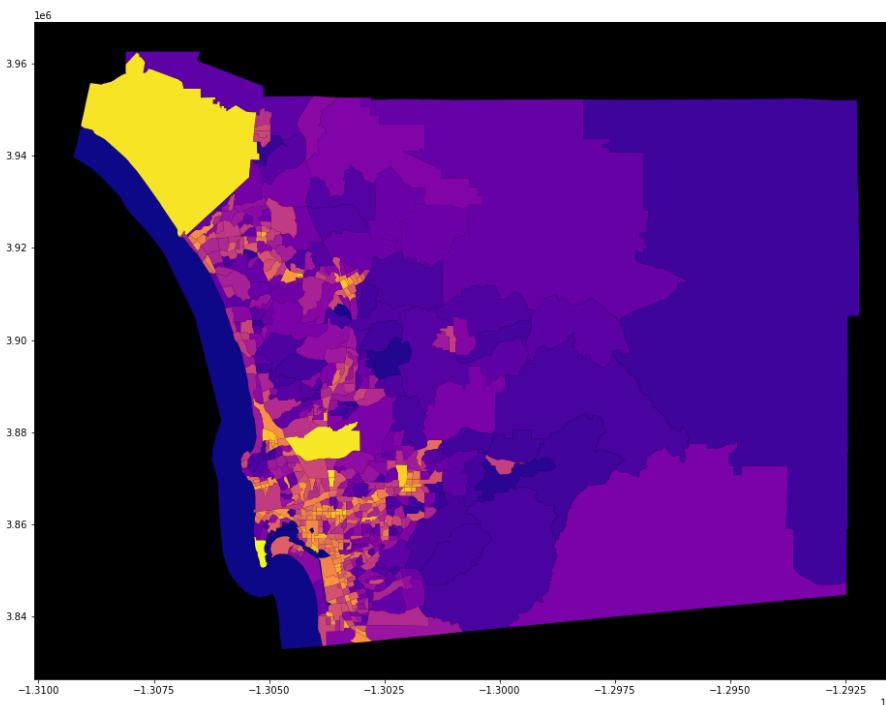
```
[1]: 1 tract_geoids = [
2     '06073000100',
3     '06073000201',
4     '06073000202',
5     '06073000300',
6     '06073000400',
7     '06073000500',
8     '06073000600',
9     '06073000700',
10    '06073000800',
11    '06073000900',
12    '06073001000',
13    '06073001100',
14    '06073001200',
15    '06073001300',
16    '06073001400',
17    '06073001500',
18    '06073001600',
19    '06073001700',
20    '06073001800',
21    '06073001900',
22    '06073002001',
23    '06073002002',
24    '06073002100',
25    '06073002201',
26    '06073002202',
27    '06073002301',
28    '06073002302',
29    '06073002401',
30    '06073002402',
31    '06073002501'
32 ]
33 db["sub_30"] = False
34 db.loc[db["GEOID"].isin(tract_geoids), "sub_30"] = True
```

```
[2]: 1 ax = db.plot(alpha=0.5, color="k")
2 db[db["sub_30"]].plot(ax=ax, color="yellow")
3 contextily.add_basemap(ax, crs=db.crs);
```



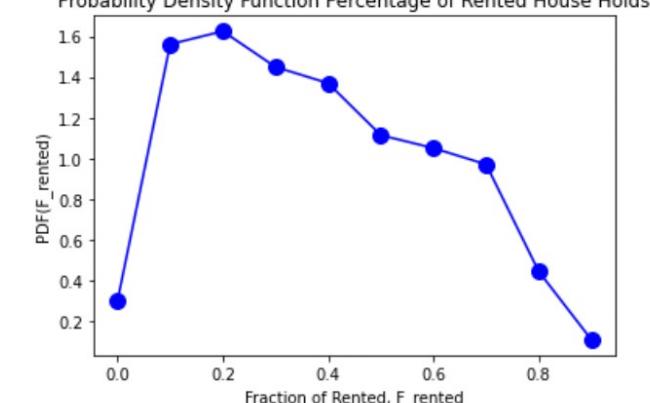
## Mapping one selected variable:

```
1 #ax = db.plot(figsize=(20, 20),cmap='plasma',legend=True)
2 #db[db[ "sub_30" ]].plot(ax=ax, color="yellow")
3 #contextily.add_basemap(ax, crs=db.crs);
4
5 f, ax = plt.subplots(1,1,figsize=(20,20))
6 db.plot('pct_rented', ax=ax, cmap='plasma',legend=True)
7 ax.set_facecolor('k')
8 plt.savefig('sandiego_map.png')
```



## Explore your results quantitatively

```
1 min(db['pct_rented'])
0.0
1 db['pct_rented'].max()
1.0
1 n1, bins1 = np.histogram(db['pct_rented'],density='True')
1 n1
array([0.30254777, 1.56050955, 1.62420382, 1.44904459, 1.36942675,
       1.11464968, 1.05095541, 0.97133758, 0.44585987, 0.11146497])
1 bins1[:-1]
array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9])
1 sum(n1*0.1)
1.0
1 plt.plot(bins1[:-1],n1,'bo-', markersize=10)
2 plt.title('Probability Density Function Percentage of Rented House Holds')
3 plt.xlabel('Fraction of Rented, F_rented')
4 plt.ylabel('PDF(F_rented)')
```



In probability theory, a **probability density function (PDF)**, or density of a continuous random variable, is a function whose value at any given sample can be interpreted as providing a *relative likelihood* that the value of the random variable would equal that sample

$$\Pr[a \leq X \leq b] = \int_a^b f_X(x) dx.$$

Probability Density curves have the following properties:

- The area under the curve always adds up to 1.
- The curve is nonnegative

```

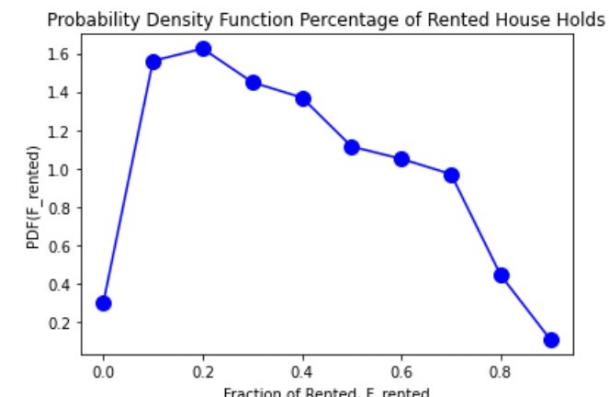
1 min(db['pct_rented'])
0.0

1 db['pct_rented'].max()
1.0

1 n1, bins1 = np.histogram(db['pct_rented'],density='True')
1 n1
array([0.30254777, 1.56050955, 1.62420382, 1.44904459, 1.36942675,
       1.11464968, 1.05095541, 0.97133758, 0.44585987, 0.11146497])

1 bins1[:-1]
array([0., 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9])
1 sum(n1*0.1)                         dx=0.1
1.0

1 plt.plot(bins1[:-1],n1,'bo-', markersize=10)
2 plt.title('Probability Density Function Percentage of Rented House Holds')
3 plt.xlabel('Fraction of Rented, F_rented')
4 plt.ylabel('PDF(F_rented)')
Text(0, 0.5, 'PDF(F_rented)')
```



# The data is saved as gpkg and json files

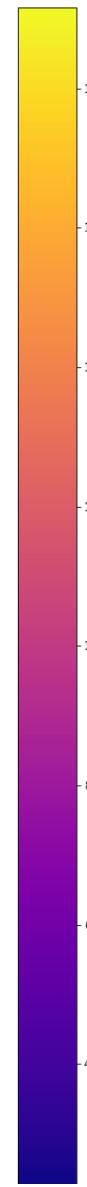
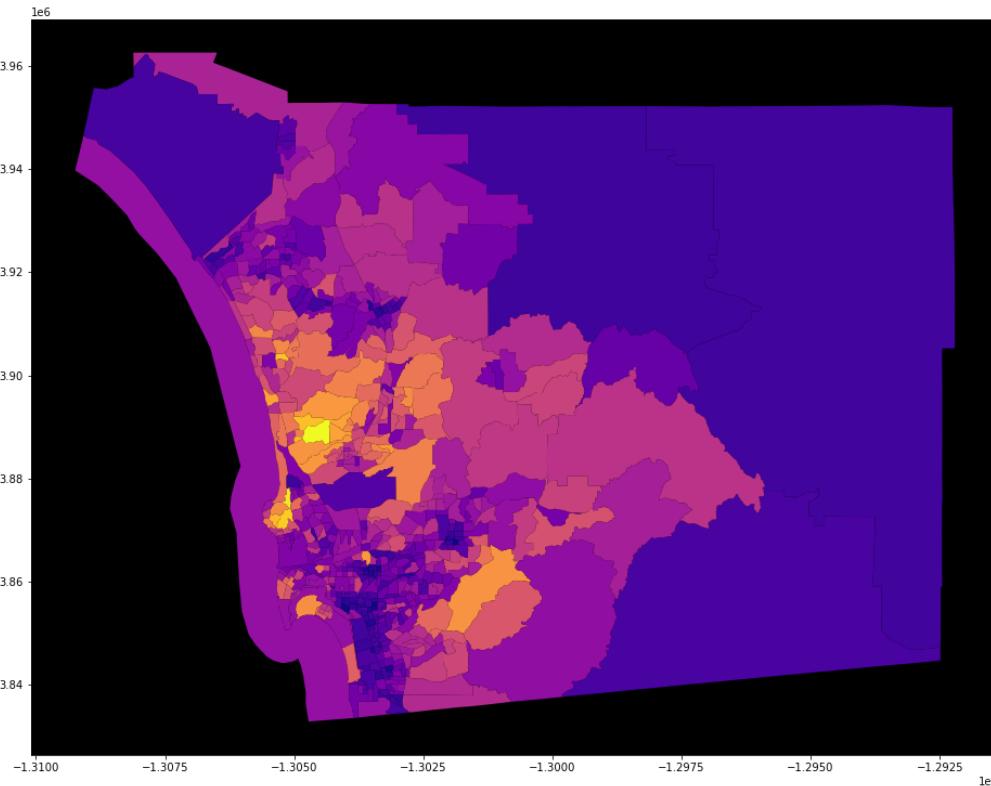
- Dataset

```
[47]: 1 ! rm -f sandiego_tracts.gpkg
2 db.to_file("sandiego_tracts.gpkg", driver="GPKG")
```

- Metadata

```
[44]: 1 ! rm -f sandiego_tracts_varnames.json
2 var_names.to_json("sandiego_tracts_varnames.json")
```

See: <https://www.gis-blog.com/geopackage-vs-shapefile/>



```
1 n2, bins2 = np.histogram(db['median_hh_income'],density='True')
```

```
1 n2
```

```
array([5.65240086e-06, 1.20584552e-05, 1.39425888e-05, 1.14932151e-05,
       7.25391444e-06, 4.14509396e-06, 2.26096034e-06, 1.69572026e-06,
       4.71033405e-07, 1.88413362e-07])
```

```
1 bins2
```

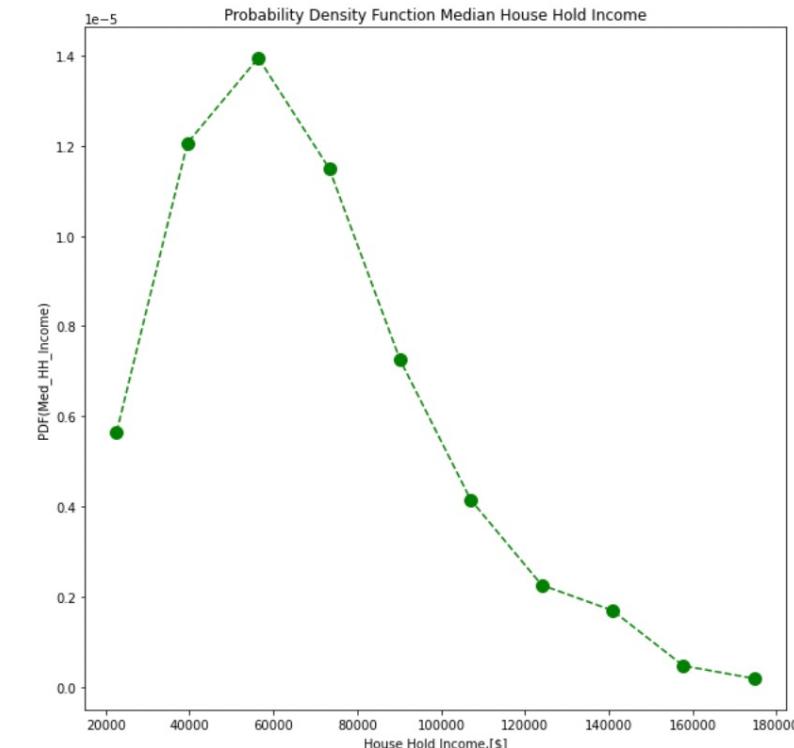
```
array([ 22614., 39516.8, 56419.6, 73322.4, 90225.2, 107128.,
       124030.8, 140933.6, 157836.4, 174739.2, 191642. ])
```

```
1 sum(n2*(bins2[1]-bins2[0]))
```

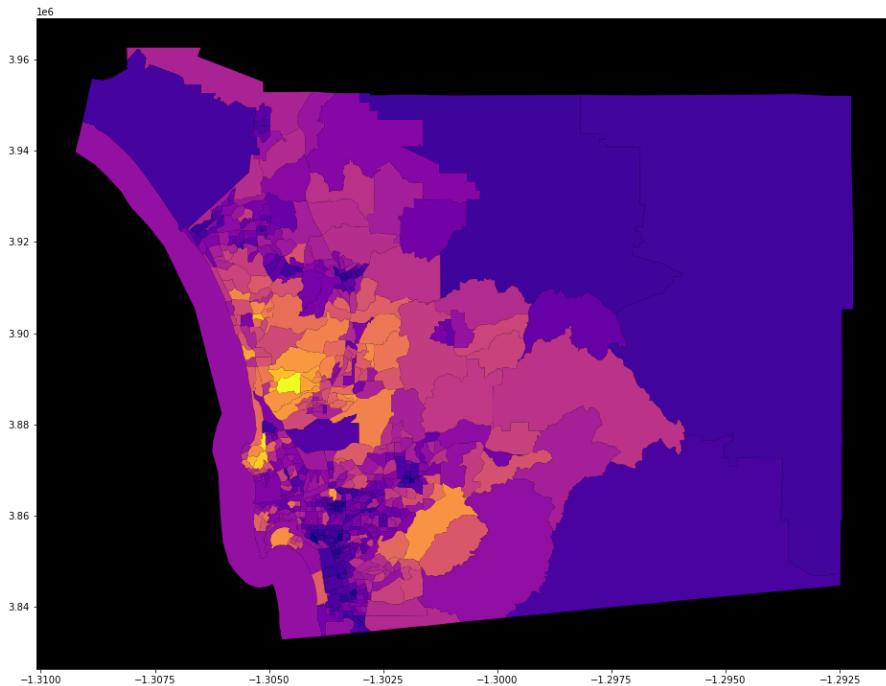
```
1.0000000000000002
```

```
1 plt.figure(figsize=(10,10))
2 plt.plot(bins2[:-1],n2,'go--', markersize=10)
3 plt.title('Probability Density Function Median House Hold Income')
4 plt.xlabel('House Hold Income,[\$]')
5 plt.ylabel('PDF(Med_HH_Income)')
```

```
Text(0, 0.5, 'PDF(Med_HH_Income)')
```



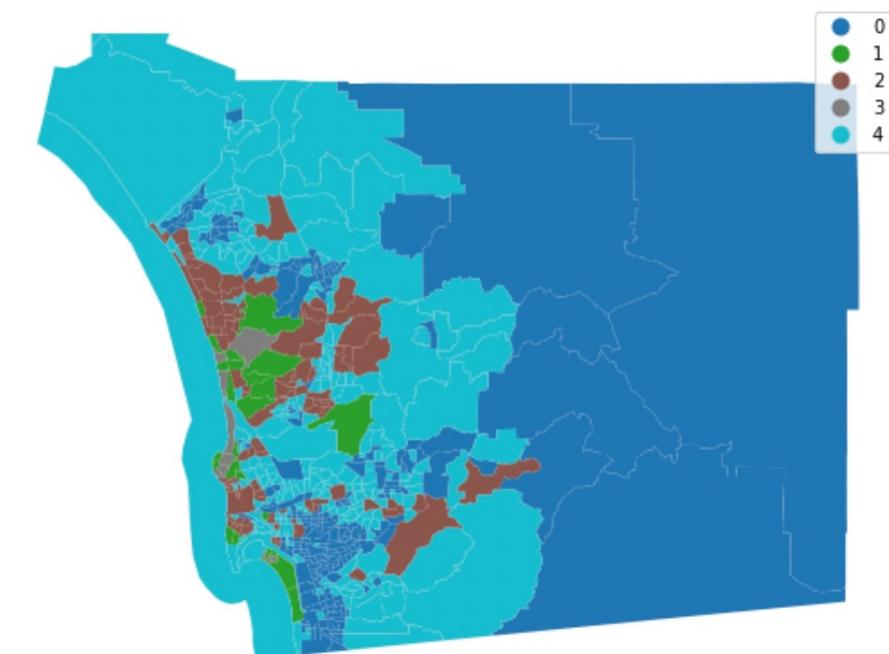
## Median Household Value



628 census tracts, 9 variables

Exercise: Name the clusters in decreasing order of Median Household Value

## 5 Clusters



k5cls	0	1	2	3	4
<b>median_house_value</b>	326728.97	1168004.00	736881.37	1876867.22	500787.57
<b>pct_white</b>	0.66	0.84	0.81	0.91	0.72
<b>pct_rented</b>	0.51	0.29	0.33	0.25	0.41
<b>pct_hh_female</b>	0.11	0.11	0.10	0.12	0.10
<b>pct_bachelor</b>	0.02	0.00	0.00	0.00	0.01
<b>median_no_rooms</b>	4.63	6.02	5.73	6.42	5.29
<b>income_gini</b>	0.40	0.47	0.43	0.52	0.40
<b>median_age</b>	34.25	44.44	41.60	50.54	37.21
<b>tt_work</b>	2187.75	2182.00	2336.28	1237.78	2539.43

# Motivation for next class

# For next class

- Finish assignment 2
- Participate in Lecture 6:  
<https://docs.google.com/presentation/d/1IxmXNjFCS1TXb7FQMB2TByHb8zHVyMBHzSzfWsTwGU/edit?usp=sharing>
- Optional Review Content of next class:  
[https://geographicdata.science/book/notebooks/10\\_clustering\\_and\\_regionlization.html](https://geographicdata.science/book/notebooks/10_clustering_and_regionlization.html)

- Clustering is a method of data analysis that draws insights from large, complex multivariate data.
- It works by finding similarities among the many dimensions in multivariate data, condensing them down into a simpler representation. Thus, through clustering, we seek to reduce the complexity of the system
- Clustering involves sorting observations into groups. For these groups to be meaningful, members of a group should be more similar to one another than they are to members of a different group.
- Each group is referred to as a *cluster* while the process of assigning objects to groups is known as *clustering*. Since a good cluster is more similar internally than it is to any other cluster, these cluster-level profiles provide a convenient shorthand to describe the original complex multivariate data.

# K-means Clustering

Given a set of observations  $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ , where each observation is a  $d$ -dimensional real vector,  $k$ -means clustering aims to partition the  $n$  observations into  $k$  ( $\leq n$ ) sets  $\mathbf{S} = \{S_1, S_2, \dots, S_k\}$  so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance). Formally, the objective is to find:

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

Vector X contains the variables of element i  
Each Set  $S_i$  has a list of elements close to their mean  $\boldsymbol{\mu}_i$

where  $\boldsymbol{\mu}_i$  is the mean of points in  $S_i$ .

This is equivalent to **minimizing the pairwise squared deviations of points in the same cluster**

This is equivalent to **maximizing the sum of squared deviations between points in different clusters (between-cluster sum of squares, BCSS)**

## Book chapter for Lectures 5 and 6:

[geographicdata.science/book/notebooks/10\\_clustering\\_andRegionalization.html](https://geographicdata.science/book/notebooks/10_clustering_andRegionalization.html)

Geographic Mapping

Global Spatial Autocorrelation

Local Spatial Autocorrelation

Point Pattern Analysis

PART III - ADVANCED TOPICS

Overview

Spatial Inequality Dynamics

Clustering & Regionalization

Spatial Regression

Spatial Feature Engineering

DATASETS

Overview

AirBnb

Airports

Brexit

Countries

GHSI

H3 Grid

Mexico

NASA DEM

San Diego Tracts

Texas

Tokyo Photographs

US County Income 1969-2017

Powered by Jupyter Book

←

Clustering & Regionalization

The world's hardest questions are complex and multi-faceted. Effective methods to learn from data recognize this. Many questions and challenges are inherently multidimensional; they are affected, shaped, and defined by many different components all acting simultaneously. In statistical terms, these processes are called *multivariate processes*, as opposed to *univariate processes*, where only a single variable acts at once. Clustering is a fundamental method of geographical analysis that draws insights from large, complex multivariate processes. It works by finding similarities among the many dimensions in a multivariate process, condensing them down into a simpler representation. Thus, through clustering, a complex and difficult to understand process is recast into a simpler one that even non-technical audiences can use.

### Introduction

Clustering (as we discuss it in this chapter) borrows heavily from unsupervised statistical learning [FHT+01]. Often, clustering involves sorting observations into groups without any prior idea about what the groups are (or, in machine learning jargon, without any labels, hence the *unsupervised* name). These groups are delineated so that members of a group should be more similar to one another than they are to members of a different group. Each group is referred to as a *cluster* while the process of assigning objects to groups is known as *clustering*. If done well, these clusters can be characterized by their *profile*, a simple summary of what members of a group are like in terms of the original multivariate phenomenon.

Since a good cluster is more similar internally than it is to any other cluster, these cluster-level profiles provide a convenient shorthand to describe the original complex multivariate phenomenon we are interested in. Observations in one group may have consistently high scores on some traits but low scores on others. The analyst only needs to look at the profile of a cluster in order to get a good sense of what all the observations in that cluster are like, instead of having to consider all of the complexities of the original multivariate process at once. Throughout data science, and particularly in geographic data science, clustering is widely used to provide insights on the (geographic) structure of complex multivariate (spatial) data.

[https://geographicdata.science/book/notebooks/10\\_clustering\\_andRegionalization.html](https://geographicdata.science/book/notebooks/10_clustering_andRegionalization.html)