The 2010 Census and Congressional Districts in 100 Simple Steps

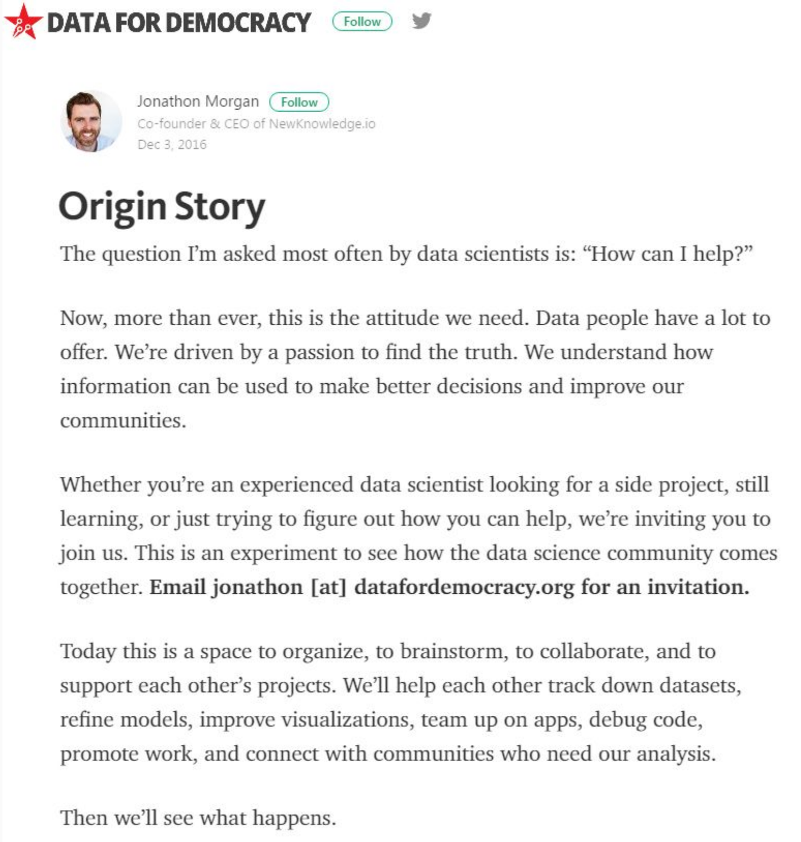
By Divya Sriram, I-Ching Wang, and Nikki Haas

Data Science W205 - Getting and Cleaning Data
University of California, Berkeley
Spring, 2017

Who are
D4D: Data for Democracy

Analyzing public data to find truth and understanding to make better decisions and improve our communities

Founded by Jonathan Morgan with over 1000 members all participating in gathering and organizing data



Jonathon Morgan
Co-founder & CEO of NewKnowledge.io
Dec 3, 2016

Origin Story

The question I'm asked most often by data scientists is: "How can I help?"

Now, more than ever, this is the attitude we need. Data people have a lot to offer. We're driven by a passion to find the truth. We understand how information can be used to make better decisions and improve our communities.

Whether you're an experienced data scientist looking for a side project, still learning, or just trying to figure out how you can help, we're inviting you to join us. This is an experiment to see how the data science community comes together. **Email jonathon [at] datafordemocracy.org for an invitation.**

Today this is a space to organize, to brainstorm, to collaborate, and to support each other's projects. We'll help each other track down datasets, refine models, improve visualizations, team up on apps, debug code, promote work, and connect with communities who need our analysis.

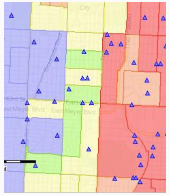
Then we'll see what happens.

Our Mission (Original)

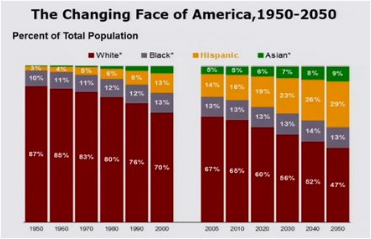
- Get the SF1 Census data for the 1990, 2000, 2010 for all states
- Groom the SF1 files to find the block-level demographics
  - A block is the smallest section of a census tract, about 11 million in the US
  - In cities, a census block is a street block
- Check the demographics to see if the most-changed congressional districts over time had strongly differing demographics from the least changed congressional districts
- Save the world

1

2



3



4

The Flaw in the Plan

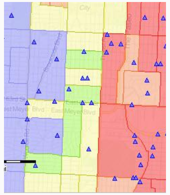
- The SF1 data for the whole nation for one census is bigger than my entire computer’s hard drive.
  - The SF1 data for one small state is 8 GBs
- The files are formatted… uniquely (more on that later)
- A Census Block != a Congressional District and the mapping does not easily exist
  - Another D4D team is working on the shapefiles for the block-level data.

Our Mission (Original)

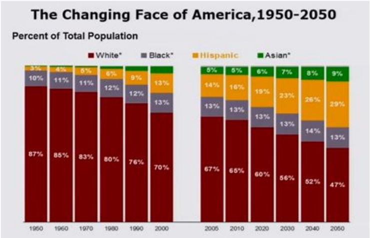
- Get the SF1 Census data for the 1990, 2000, 2010 for all states
- Groom the SF1 files to find the block-level demographics
  - A block is the smallest section of a census tract, about 11 million in the US
  - In cities, a census block is a street block
- Check the demographics to see if the most-changed congressional districts over time had strongly differing demographics from the least changed congressional districts over time
- Save the world

1

2



3



4

Our Mission (Revised)

- Get the SF1 Census data for 2010 for top 3 gerrymandered states as defined by D4D

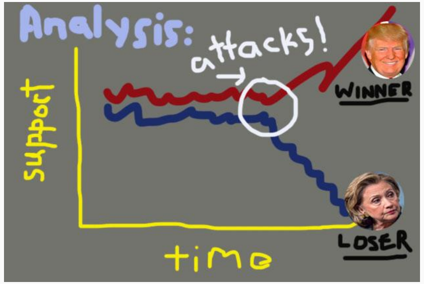
- Groom the SF1 files to find county-level demographics
- Check the demographics to see if the counties in the contentious congressional districts had strongly differing demographics from the mean demographics in that state
- Save the world

1

2

3

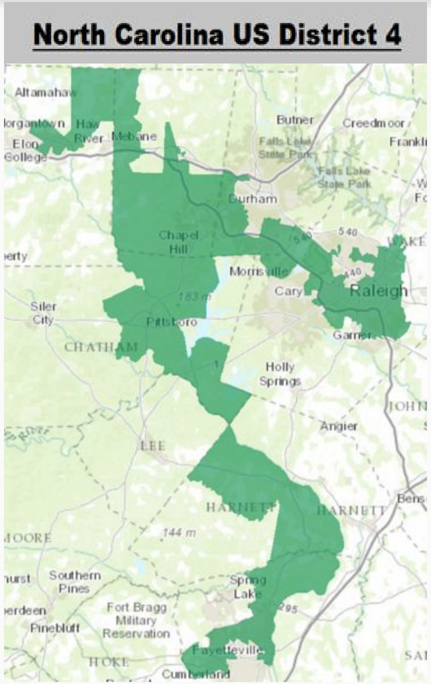
4



5

What is a Contentious Congressional District?

- A CD that was part of a lawsuit
- A CD with a very odd shape
- A CD with numerous complains of gerrymandering



North Carolina US District 4

Steps 1 - 80: Getting and Cleaning Data

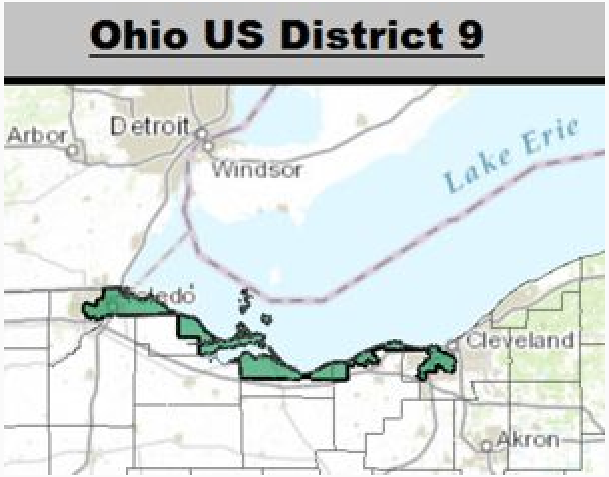
Step 1 - 5 : Finding the right 2010 Census summary data

Step 6-11: Identify top 3 accused Gerrymandering states and Congressional Districts

Steps 12- 80: Get that data in a usable format for ~~HDFS/Hive/Spark~~ Pandas

Top 3 Gerrymandering States

- Gerrymandering: manipulate the boundaries of an electoral constituency so as to favor one party or class
- Selected States: NC, TN, OH
    - Identified by D4D
    - States all have recently had or currently have Supreme Court suits filed due to gerrymandering claims



Steps 12-80:
Deciphering tables

“Wonderful” 730 page PDF document indicating all column information for almost 50 tfl files with no header information

Two options for the header: numerical (P0#######) or the actual text description. We put both into a header file.

45 total tables filtered

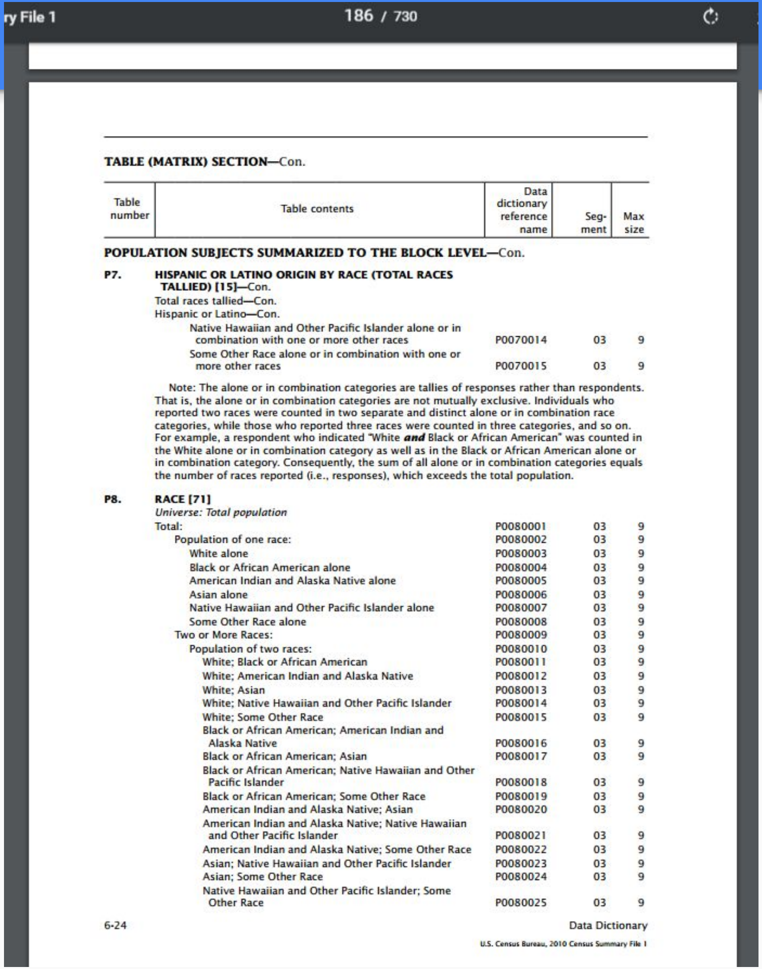


TABLE (MATRIX) SECTION—Con.

Table number	Table contents	Data dictionary reference name	Seg-ment	Max size
	POPULATION SUBJECTS SUMMARIZED TO THE BLOCK LEVEL—Con.			
P7.	**HISPANIC OR LATINO ORIGIN BY RACE (TOTAL RACES TALLIED) [15]**—Con.			
	Total races tallied—Con.			
	Hispanic or Latino—Con.			
	Native Hawaiian and Other Pacific Islander alone or in combination with one or more other races	P0070014	03	9
	Some Other Race alone or in combination with one or more other races	P0070015	03	9
P8.	**RACE [71]**			
	Universe: Total population			
	Total:	P0080001	03	9
	Population of one race:	P0080002	03	9
	White alone	P0080003	03	9
	Black or African American alone	P0080004	03	9
	American Indian and Alaska Native alone	P0080005	03	9
	Asian alone	P0080006	03	9
	Native Hawaiian and Other Pacific Islander alone	P0080007	03	9
	Some Other Race alone	P0080008	03	9
	Two or More Races:	P0080009	03	9
	Population of two races:	P0080010	03	9
	White; Black or African American	P0080011	03	9
	White; American Indian and Alaska Native	P0080012	03	9
	White; Asian	P0080013	03	9
	White; Native Hawaiian and Other Pacific Islander	P0080014	03	9
	White; Some Other Race	P0080015	03	9
	Black or African American; American Indian and Alaska Native	P0080016	03	9
	Black or African American; Asian	P0080017	03	9
	Black or African American; Native Hawaiian and Other Pacific Islander	P0080018	03	9
	Black or African American; Some Other Race	P0080019	03	9
	American Indian and Alaska Native; Asian	P0080020	03	9
	American Indian and Alaska Native; Native Hawaiian and Other Pacific Islander	P0080021	03	9
	American Indian and Alaska Native; Some Other Race	P0080022	03	9
	Asian; Native Hawaiian and Other Pacific Islander	P0080023	03	9
	Asian; Some Other Race	P0080024	03	9
	Native Hawaiian and Other Pacific Islander; Some Other Race	P0080025	03	9

Note: The alone or in combination categories are tallies of responses rather than respondents. That is, the alone or in combination categories are not mutually exclusive. Individuals who reported two races were counted in two separate and distinct alone or in combination race categories, while those who reported three races were counted in three categories, and so on. For example, a respondent who indicated "White ***and*** Black or African American" was counted in the White alone or in combination category as well as in the Black or African American alone or in combination category. Consequently, the sum of all alone or in combination categories equals the number of races reported (i.e., responses), which exceeds the total population.

6-24 Data Dictionary

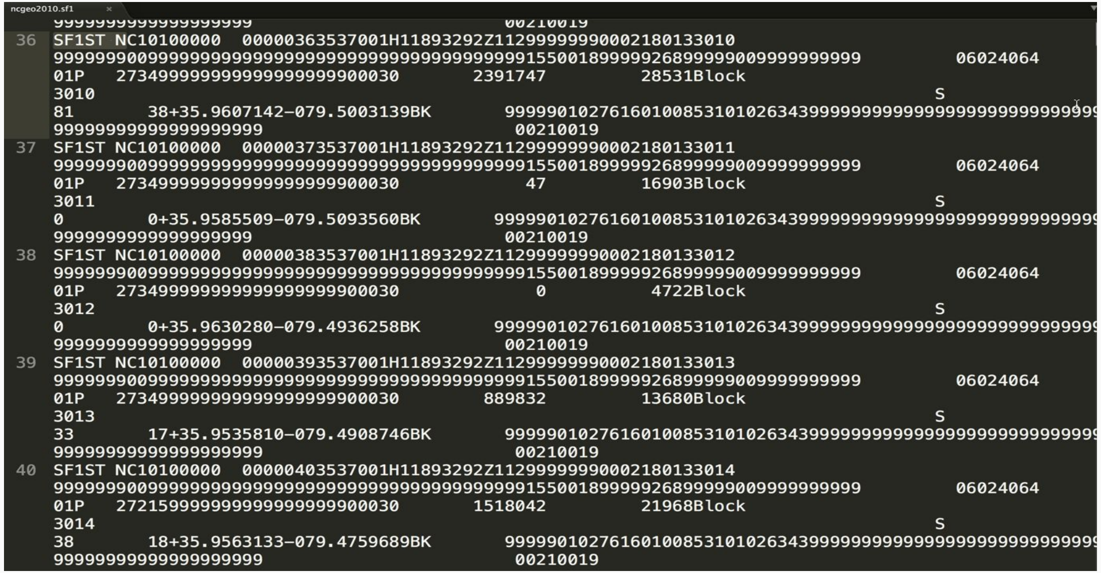
U.S. Census Bureau, 2010 Census Summary File 1

Steps 12 - 80: Getting the Row Names

SF1 data comes with LOGRECNO codes.

LOGRECNO codes correspond to geo-codes.

Geo-code to LOGRECNO code file formatted like so:



```
9999999999999999999                         00210019
SF1ST NC10100000  00000363537001H11893292Z112999999990002180133010
9999990099999999999999999999999999999999999999915500189999926899999009999999999         06024064
01P   27349999999999999999900030       2391747          28531Block
3010                                                                                  S
81       38+35.9607142-079.5003139BK        999901027616010085310102634399999999999999999999999999999
99999999999999999999                         00210019
SF1ST NC10100000  00000373537001H11893292Z112999999990002180133011
9999990099999999999999999999999999999999999999915500189999926899999009999999999         06024064
01P   27349999999999999999900030            47          16903Block
3011                                                                                  S
0        0+35.9585509-079.5093560BK        9999010276160100853101026343999999999999999999999999999999
9999999999999999999                         00210019
SF1ST NC10100000  00000383537001H11893292Z112999999990002180133012
9999990099999999999999999999999999999999999999915500189999926899999009999999999         06024064
01P   27349999999999999999900030             0           4722Block
3012                                                                                  S
0        0+35.9630280-079.4936258BK        9999010276160100853101026343999999999999999999999999999999
9999999999999999999                         00210019
SF1ST NC10100000  00000393537001H11893292Z112999999990002180133013
9999990099999999999999999999999999999999999999915500189999926899999009999999999         06024064
01P   27349999999999999999900030        889832          13680Block
3013                                                                                  S
33       17+35.9535810-079.4908746BK        999901027616010085310102634399999999999999999999999999999
99999999999999999999                         00210019
SF1ST NC10100000  00000403537001H11893292Z112999999990002180133014
9999990099999999999999999999999999999999999999915500189999926899999009999999999         06024064
01P   27215999999999999999900030       1518042          21968Block
3014                                                                                  S
38       18+35.9563133-079.4759689BK        999901027616010085310102634399999999999999999999999999999
99999999999999999999                         00210019
```

**We were 30 hours into the project when we realized this*

Steps 12 - 80: Usable Format for ~~HDFS~~ Pandas

Was this an error? NO! It's a fixed column width (variable columns) ASCII file

Thank you Professor Mack of University of Delaware, for this decoder ring

Curse you, Prof Mack for doing this ONLY for Delaware

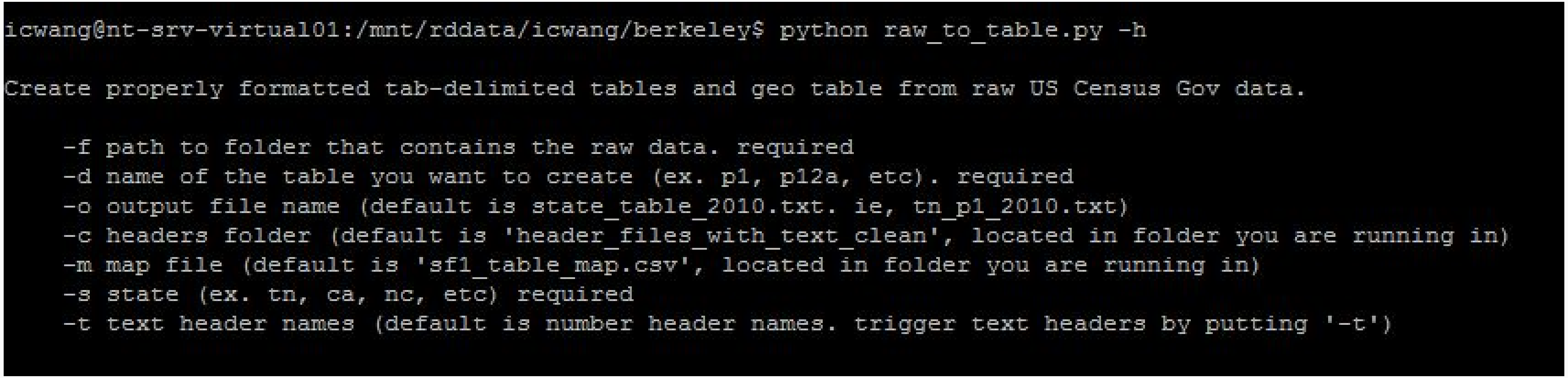
Ultimately we used pandas: pd.read_fwf(file, widths = [list])

Final required files created files

- 2 row header files for each table: first row with # ID, second row with text ID (manually created)
- Raw data (downloaded from US Census and unzipped)
- Map to find which columns from which raw data file to use to generate the correct tables indicated in the 730 pg document (manually created)
- Map to find which rows from the newly generated tables to use that indicate gerrymandering counties (manually created)

Steps 81-100: Preparing all Tables

raw_to_table.py will prepare all the above steps

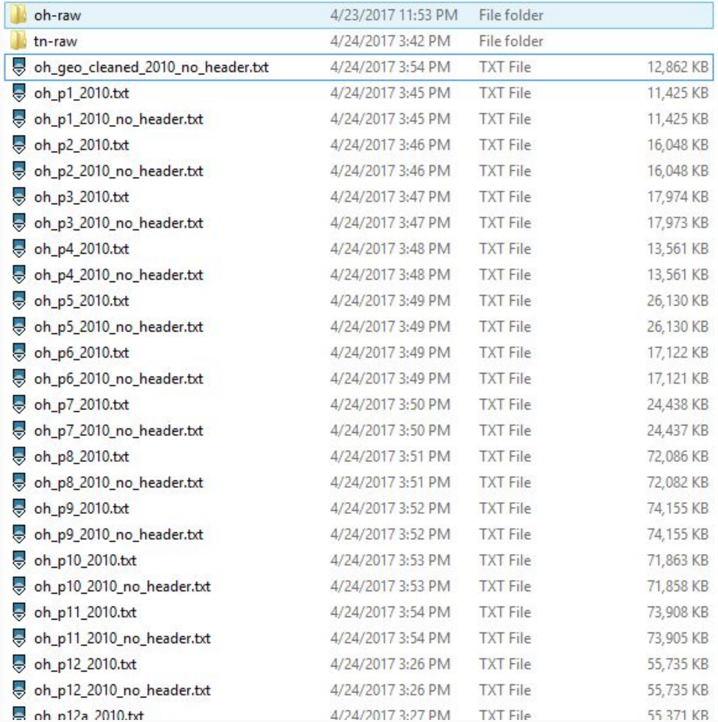


```
icwang@nt-srv-virtual01:/mnt/rddata/icwang/berkeley$ python raw_to_table.py -h

Create properly formatted tab-delimited tables and geo table from raw US Census Gov data.

    -f path to folder that contains the raw data. required
    -d name of the table you want to create (ex. p1, p12a, etc). required
    -o output file name (default is state_table_2010.txt. ie, tn_p1_2010.txt)
    -c headers folder (default is 'header_files_with_text_clean', located in folder you are running in)
    -m map file (default is 'sf1_table_map.csv', located in folder you are running in)
    -s state (ex. tn, ca, nc, etc) required
    -t text header names (default is number header names. trigger text headers by putting '-t')
```

Output from raw_to_table.py script



oh-raw	4/23/2017 11:53 PM	File folder	
tn-raw	4/24/2017 3:42 PM	File folder	
oh_geo_cleaned_2010_no_header.txt	4/24/2017 3:54 PM	TXT File	12,862 KB
oh_p1_2010.txt	4/24/2017 3:45 PM	TXT File	11,425 KB
oh_p1_2010_no_header.txt	4/24/2017 3:45 PM	TXT File	11,425 KB
oh_p2_2010.txt	4/24/2017 3:46 PM	TXT File	16,048 KB
oh_p2_2010_no_header.txt	4/24/2017 3:46 PM	TXT File	16,048 KB
oh_p3_2010.txt	4/24/2017 3:47 PM	TXT File	17,974 KB
oh_p3_2010_no_header.txt	4/24/2017 3:47 PM	TXT File	17,973 KB
oh_p4_2010.txt	4/24/2017 3:48 PM	TXT File	13,561 KB
oh_p4_2010_no_header.txt	4/24/2017 3:48 PM	TXT File	13,561 KB
oh_p5_2010.txt	4/24/2017 3:49 PM	TXT File	26,130 KB
oh_p5_2010_no_header.txt	4/24/2017 3:49 PM	TXT File	26,130 KB
oh_p6_2010.txt	4/24/2017 3:49 PM	TXT File	17,122 KB
oh_p6_2010_no_header.txt	4/24/2017 3:49 PM	TXT File	17,121 KB
oh_p7_2010.txt	4/24/2017 3:50 PM	TXT File	24,438 KB
oh_p7_2010_no_header.txt	4/24/2017 3:50 PM	TXT File	24,437 KB
oh_p8_2010.txt	4/24/2017 3:51 PM	TXT File	72,086 KB
oh_p8_2010_no_header.txt	4/24/2017 3:51 PM	TXT File	72,082 KB
oh_p9_2010.txt	4/24/2017 3:52 PM	TXT File	74,155 KB
oh_p9_2010_no_header.txt	4/24/2017 3:52 PM	TXT File	74,155 KB
oh_p10_2010.txt	4/24/2017 3:53 PM	TXT File	71,863 KB
oh_p10_2010_no_header.txt	4/24/2017 3:53 PM	TXT File	71,858 KB
oh_p11_2010.txt	4/24/2017 3:54 PM	TXT File	73,908 KB
oh_p11_2010_no_header.txt	4/24/2017 3:54 PM	TXT File	73,905 KB
oh_p12_2010.txt	4/24/2017 3:26 PM	TXT File	55,735 KB
oh_p12_2010_no_header.txt	4/24/2017 3:26 PM	TXT File	55,735 KB
oh_p12a_2010.txt	4/24/2017 3:27 PM	TXT File	55,371 KB

Output will be a table with the header included and a version without the header to be automatically compatible with PostgreSQL and Hive

Output from raw_to_table all you need to go into gerry_analysis.py

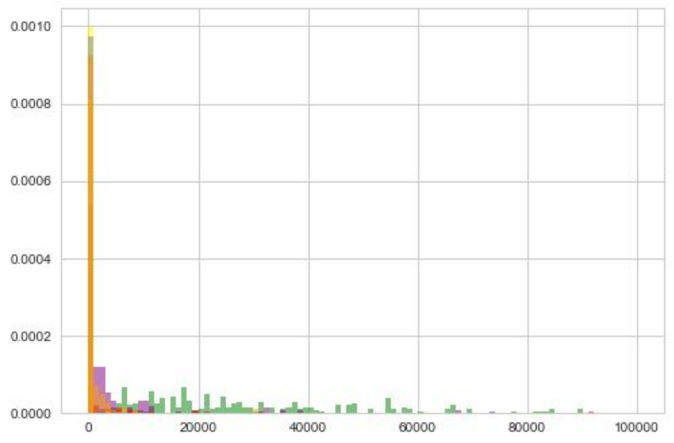
Steps 81 -100: Processing all states

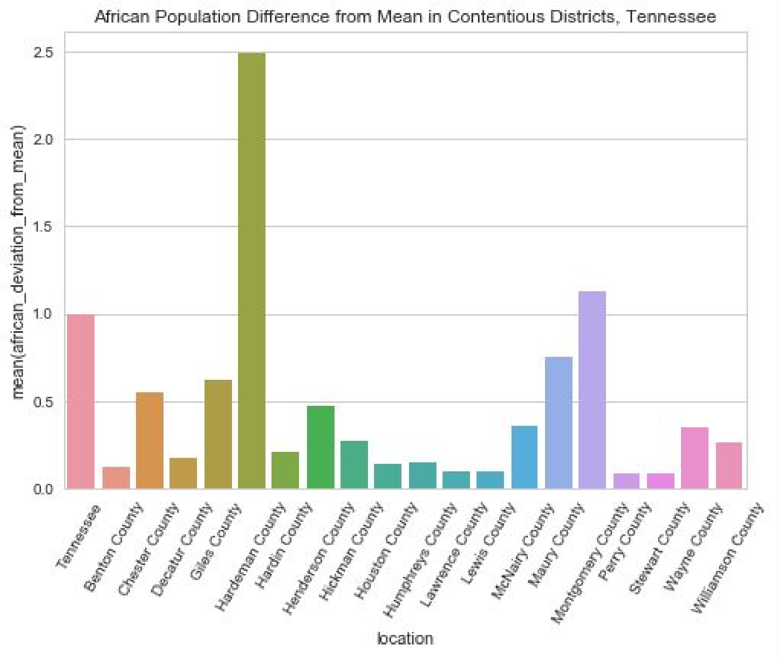
Now it’s automatable and scalable!

- Loop the script to do multiple tables in a batch
- Loop through states to do multiple states
- Space conservation

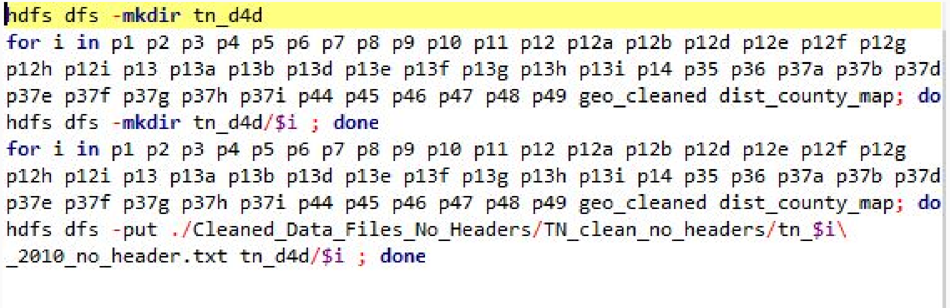
Steps 81-100: Preliminary Look at Raw Data

Ipython notebook exploration of data to see what the data looks like

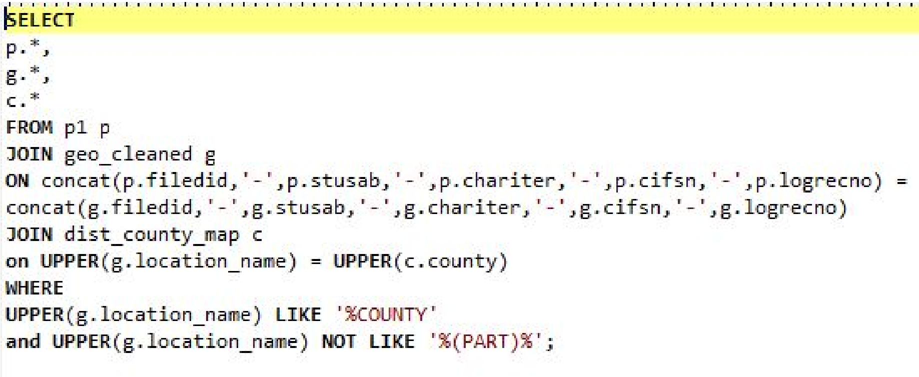




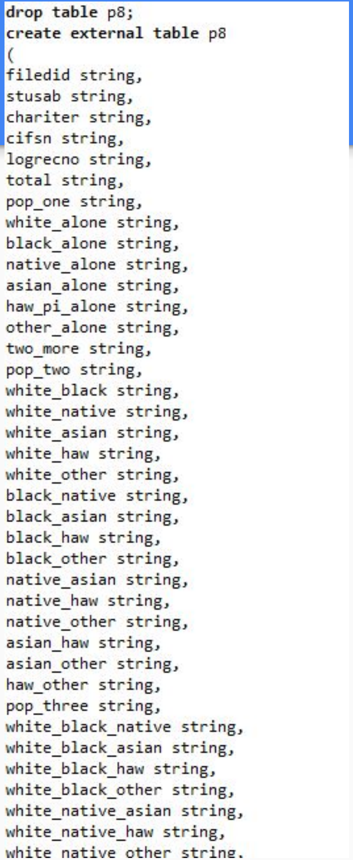
Steps 81-100: Load into ~~PostgreSQL and Hive queries~~



```
hdfs dfs -mkdir tn_d4d
for i in p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p12a p12b p12d p12e p12f p12g
p12h p12i p13 p13a p13b p13d p13e p13f p13g p13h p13i p14 p35 p36 p37a p37b p37d
p37e p37f p37g p37h p37i p44 p45 p46 p47 p48 p49 geo_cleaned dist_county_map; do
hdfs dfs -mkdir tn_d4d/$i ; done
for i in p1 p2 p3 p4 p5 p6 p7 p8 p9 p10 p11 p12 p12a p12b p12d p12e p12f p12g
p12h p12i p13 p13a p13b p13d p13e p13f p13g p13h p13i p14 p35 p36 p37a p37b p37d
p37e p37f p37g p37h p37i p44 p45 p46 p47 p48 p49 geo_cleaned dist_county_map; do
hdfs dfs -put ./Cleaned_Data_Files_No_Headers/TN_clean_no_headers/tn_$i\
_2010_no_header.txt tn_d4d/$i ; done
```



```
SELECT
p.*,
g.*,
c.*
FROM p1 p
JOIN geo_cleaned g
ON concat(p.filedid,'-',p.stusab,'-',p.chariter,'-',p.cifsn,'-',p.logrecno) =
concat(g.filedid,'-',g.stusab,'-',g.chariter,'-',g.cifsn,'-',g.logrecno)
JOIN dist_county_map c
on UPPER(g.location_name) = UPPER(c.county)
WHERE
UPPER(g.location_name) LIKE '%COUNTY'
and UPPER(g.location_name) NOT LIKE '%(PART)%';
```



```
drop table p8;
create external table p8
(
filedid string,
stusab string,
chariter string,
cifsn string,
logrecno string,
total string,
pop_one string,
white_alone string,
black_alone string,
native_alone string,
asian_alone string,
haw_pi_alone string,
other_alone string,
two_more string,
pop_two string,
white_black string,
white_native string,
white_asian string,
white_haw string,
white_other string,
black_native string,
black_asian string,
black_haw string,
black_other string,
native_asian string,
native_haw string,
native_other string,
asian_haw string,
asian_other string,
haw_other string,
pop_three string,
white_black_native string,
white_black_asian string,
white_black_haw string,
white_black_other string,
white_native_asian string,
white_native_haw string,
white_native_other string,
```

Backwards Paddle to Pandas...

A quick look at the table shows a problem from the beginning: non-unique identifiers.

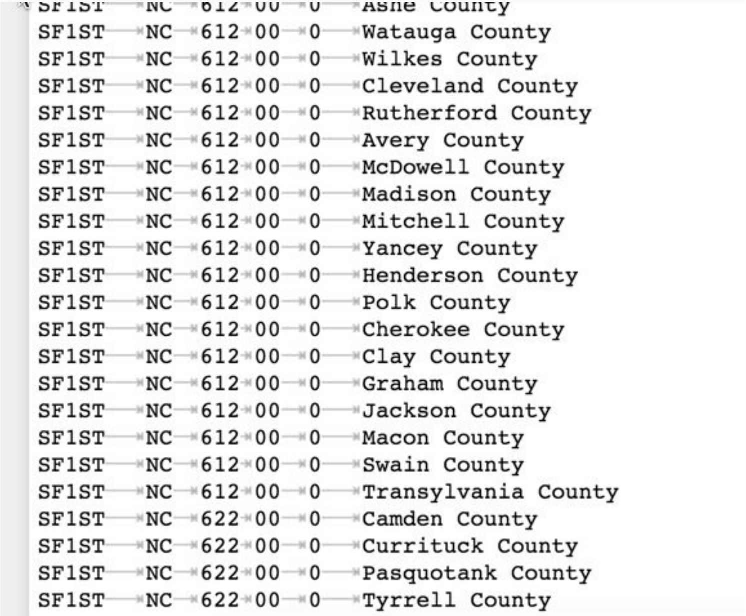
Deviation from original plan: use Pandas instead when we realized the 5 columns that were to be the keys are **not unique at the county level.**

> The key had to be the row index in Pandas

Why did we have to use Pandas?

The LOGRENCO field is not unique for county data.

Pandas allows for joining on the row index, but Hive and other relational databases do not.

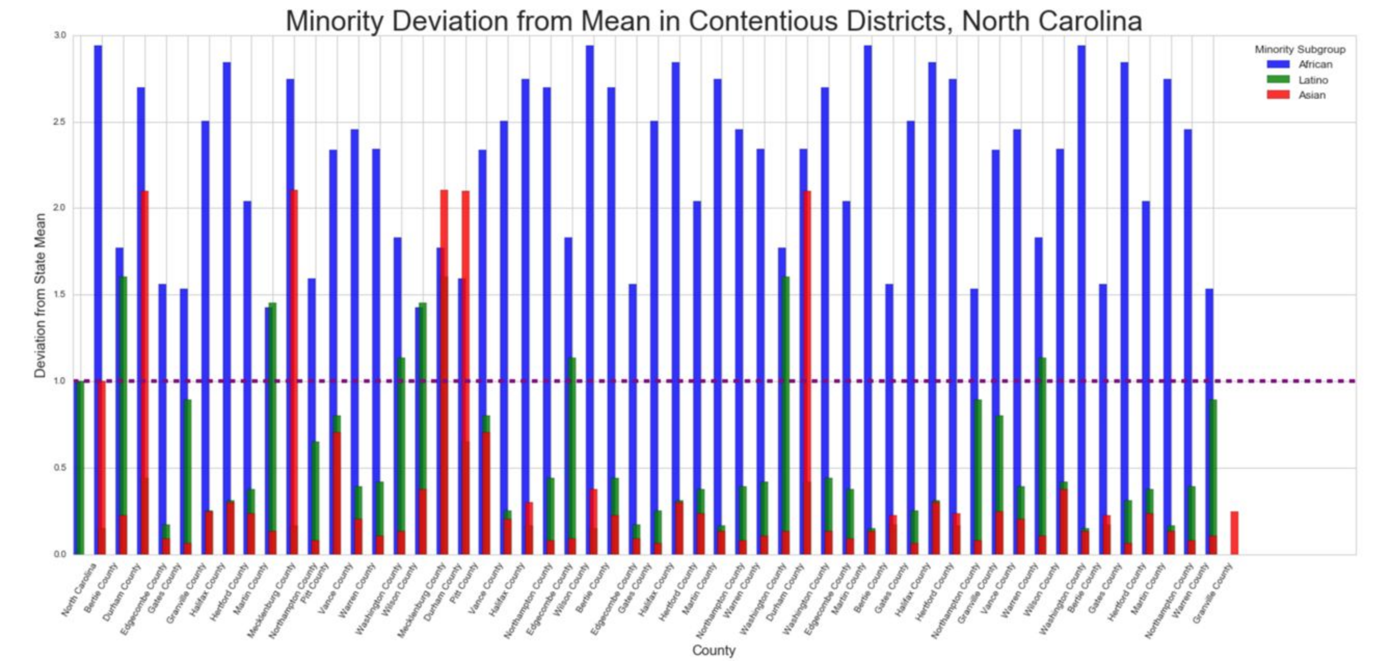


```
SF1ST   NC  612 00  0   Ashe County
SF1ST   NC  612 00  0   Watauga County
SF1ST   NC  612 00  0   Wilkes County
SF1ST   NC  612 00  0   Cleveland County
SF1ST   NC  612 00  0   Rutherford County
SF1ST   NC  612 00  0   Avery County
SF1ST   NC  612 00  0   McDowell County
SF1ST   NC  612 00  0   Madison County
SF1ST   NC  612 00  0   Mitchell County
SF1ST   NC  612 00  0   Yancey County
SF1ST   NC  612 00  0   Henderson County
SF1ST   NC  612 00  0   Polk County
SF1ST   NC  612 00  0   Cherokee County
SF1ST   NC  612 00  0   Clay County
SF1ST   NC  612 00  0   Graham County
SF1ST   NC  612 00  0   Jackson County
SF1ST   NC  612 00  0   Macon County
SF1ST   NC  612 00  0   Swain County
SF1ST   NC  612 00  0   Transylvania County
SF1ST   NC  622 00  0   Camden County
SF1ST   NC  622 00  0   Currituck County
SF1ST   NC  622 00  0   Pasquotank County
SF1ST   NC  622 00  0   Tyrrell County
```

Steps 81-100: Analysis and Exploration

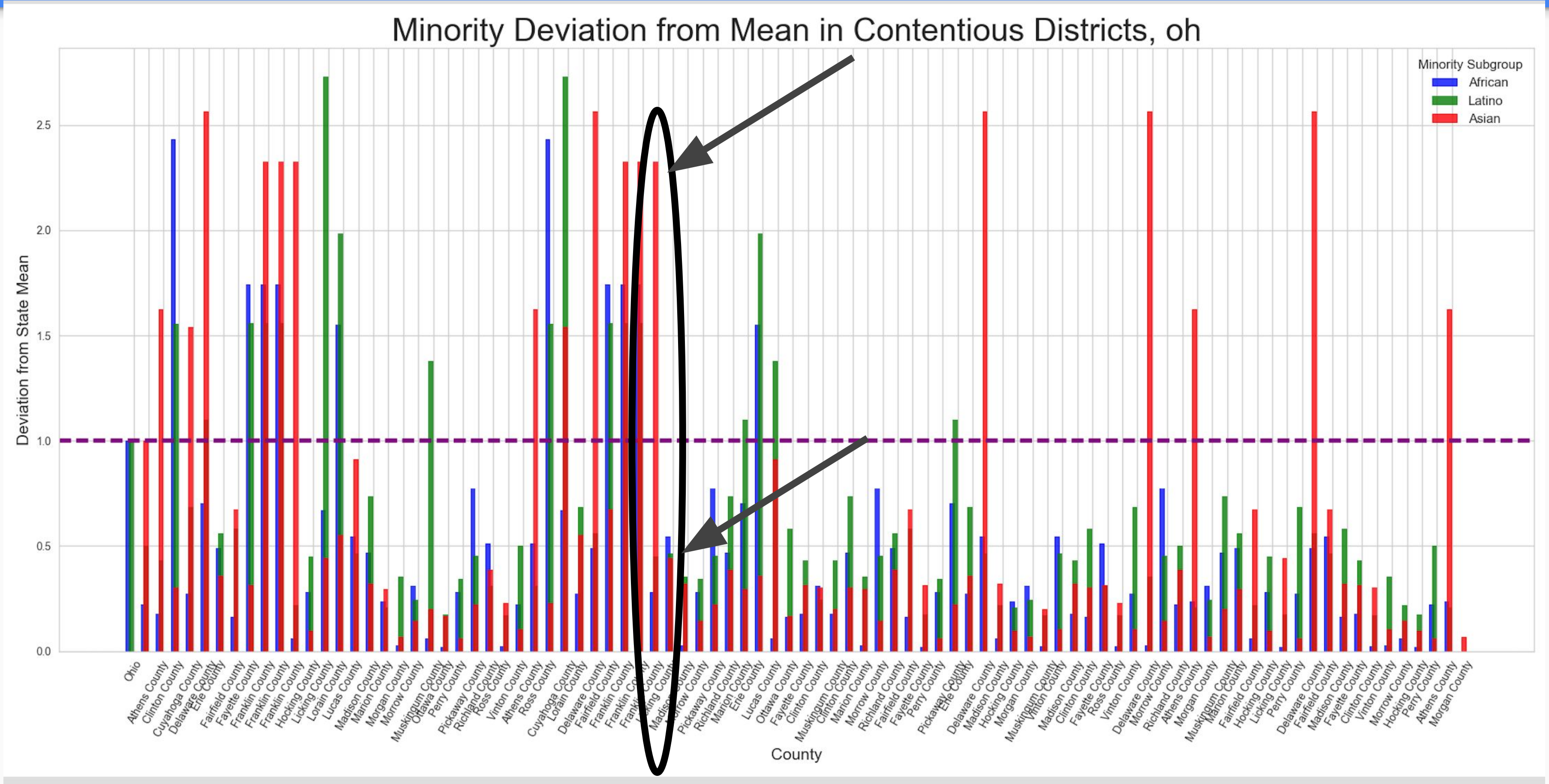
Q: Are the demographic breakdowns for the contentious congressional districts similar to the state's demographics as a whole?

Steps 81-100: North Carolina

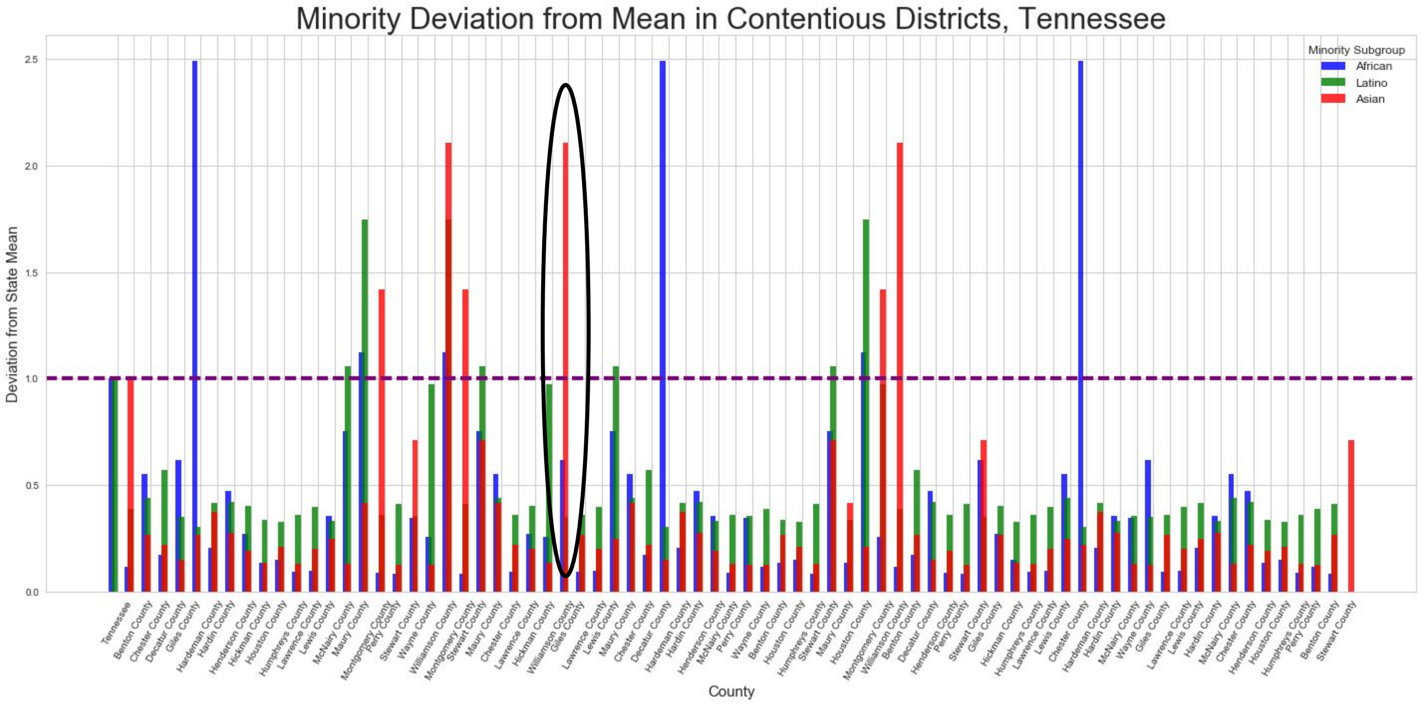


It appears so for North Carolina; for example Washington County is within a contentious Congressional District and has 3 times the proportion of African Americans than the state's mean.

Steps 81-100: Ohio



Steps 81-100: TN

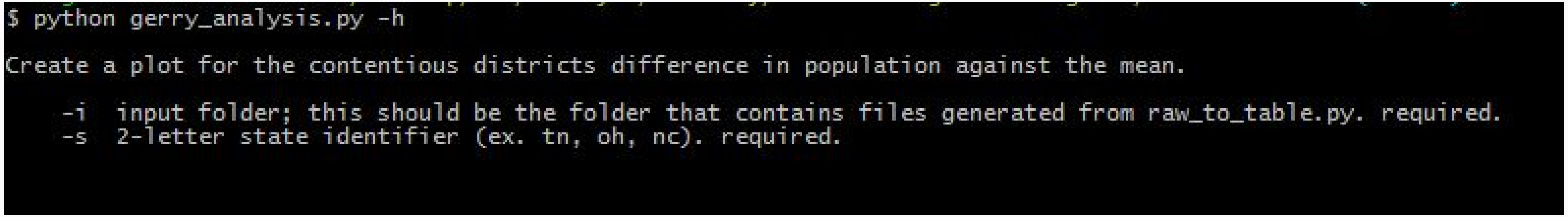


Yes, we see some differences

Majority of counties have very low minority subgroup representation compared to the mean

Circled is Williamson County - 2x the median for the 7th district

Script for gerry_analysis

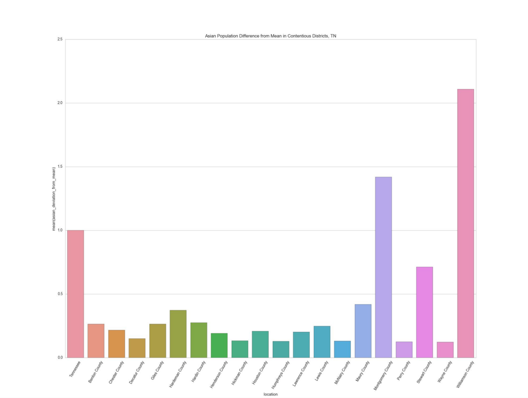


```
$ python gerry_analysis.py -h

Create a plot for the contentious districts difference in population against the mean.

    -i  input folder; this should be the folder that contains files generated from raw_to_table.py. required.
    -s  2-letter state identifier (ex. tn, oh, nc). required.
```

Easily expandable to include additional plots as needed



Back to D4D

All scripts used for any state - ready for any other D4D analysis

Analysis submitted back to D4D for the 3 states

Conclusion

Never volunteer to work on government data