Using Social Media and U.S. Census Data to Map Gender Imbalance

Thomaz Lago Santana

November 5, 2015

Introduction

- Thomaz spelled with a “Z”
- I’m from Brazil.
- Florida International University (Miami, FL)
  - B.S. in Physics
  - M.S. in Engineering Management
  - B.S. in Mechanical Engineering (70% completed)
- I love taking classes on Coursera!
  - Completed 5 out of 9 towards a program in Data Science

Overview

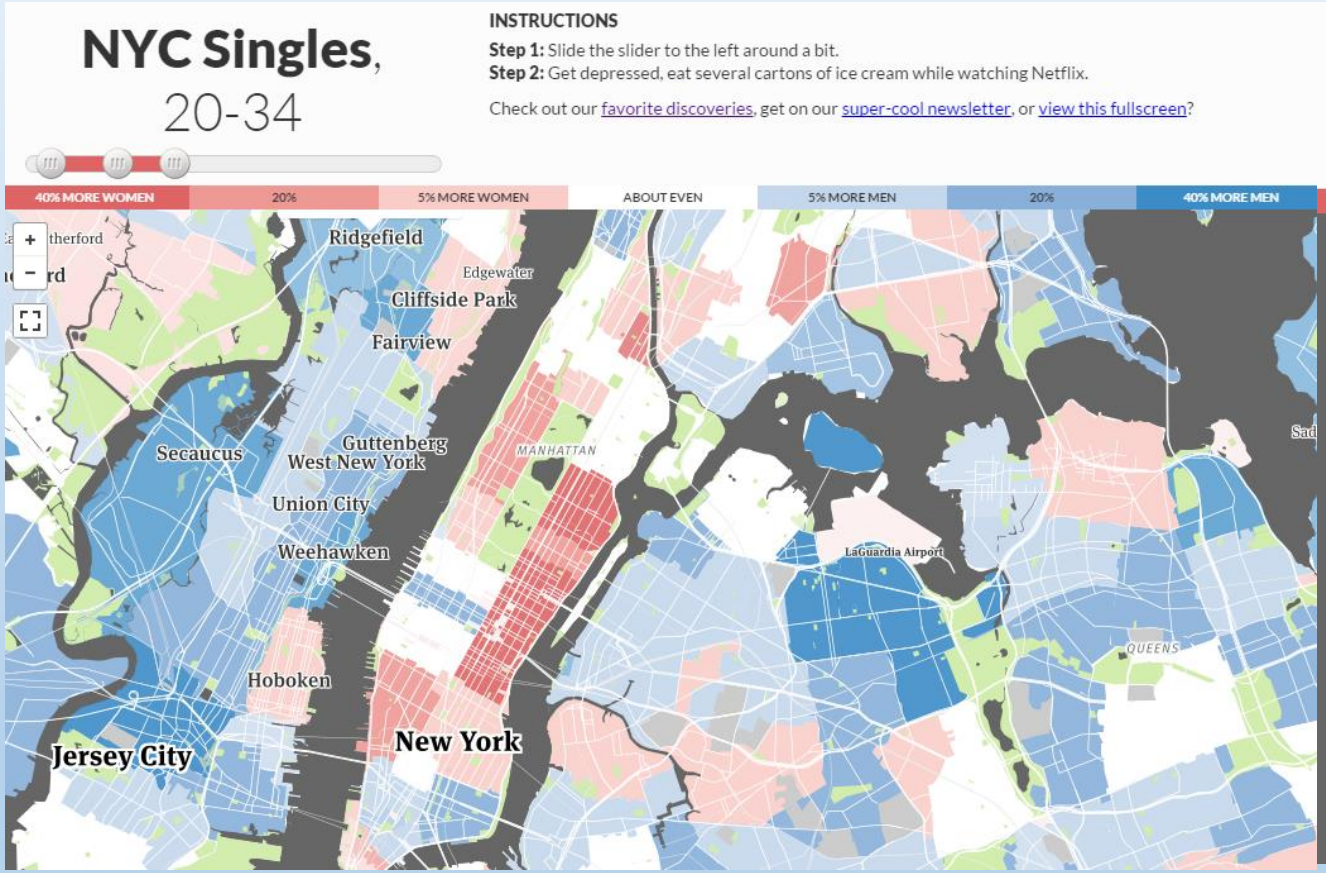
- Why this project?
- Data sources
- Massaging data (cleaning data)
- Data visualization
- Discussion

Why This Project?

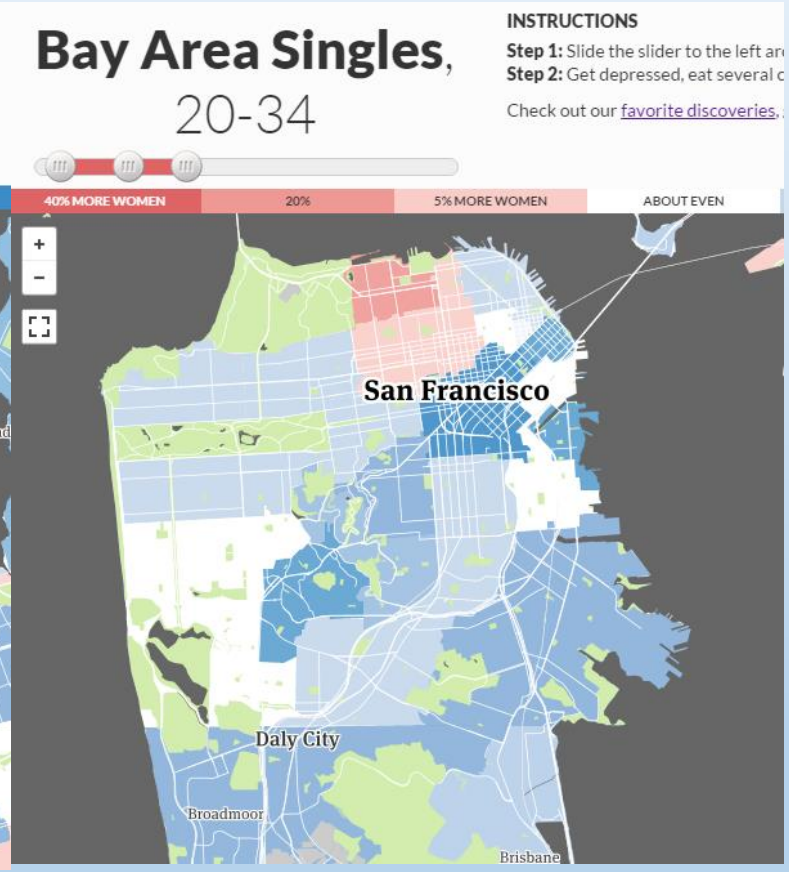
- Gender imbalance may result in the threat of social unrest, especially in the case of an excess of low-status young males unable to find spouses.[1]
- Economic factors and specific industries create a gender imbalance.[2]
- Also, I was curious.

[1] *Bare Branches: The Security Implications of Asia's Surplus Male Population*, Valerie M. Hudson and Andrea M. den Boer (2004)
[2] Florida, Richard (March 30, 2008). "A singles map of the United States of America". The Boston Globe.

Inspiration



http://visualizing.nyc/nyc-zip-codes-singles-map/



http://visualizing.nyc/bay-area-zip-codes-singles-map

What is Different About This Project?

- All reports use U.S. Census data to calculate the gender ratio.
- Many people are in long term relationships without being married or chose to not be in a relationship.
- The Census data does not consider sexual orientation.
- Public profiles from dating sites can fill these gaps.

Data Sources

- U.S. Census (2013 acs5 dataset)
- Public profiles on dating websites
- U.S. Gazetteer

U.S. Census API

- The API only provides aggregated data.
  - Micro data is available at https://usa.ipums.org/usa/
- There are 2,432 variables.
  - E.g. B12002_005E – Male, Never Married, 18-19 years old
- Example API query
  - http://api.census.gov/data/2013/acs5?get=NAME,B01001_001E&for=state:*&key=YOUR_KEY_GOES_HERE
- Federal Information Processing Standard (FIPS)
  - E.g. Oregon = 41, Washington County = 067

Python API

```python
import urllib.request
import functools
import ast
class Census:
    def __init__(self, key):
        self.key = key

    def get(self, fields, geo, year=2013, dataset='acs5'):
        fields = [','.join(fields)]
        base_url = 'http://api.census.gov/data/%s/%s?key=%s&get=' % (str(year), dataset, self.key)
        query = fields
        for item in geo:
            query.append(item)
        add_url = '&'.join(query)
        url = base_url + add_url
        req = urllib.request.Request(url)
        response = urllib.request.urlopen(req)
        return ast.literal_eval(response.read().decode('utf8'))
```


```
[["NAME","B01001_001E","state"],
["Alabama","4799277","01"],
["Alaska","720316","02"],
["Arizona","6479703","04"],
["Arkansas","2933369","05"],
["California","37659181","06"],
["Colorado","5119329","08"],
["Connecticut","3583561","09"],
["Delaware","908446","10"],
["District of Columbia","619371","11"],
["Florida","19091156","12"],
["Georgia","9810417","13"],
["Hawaii","1376298","15"],
["Idaho","1583364","16"],
["Illinois","12848554","17"],
["Indiana","6514861","18"],
["Iowa","3062553","19"],
["Kansas","2868107","20"],
["Kentucky","4361333","21"],
["Louisiana","4567968","22"],
["Maine","1328320","23"],
["Maryland","5834299","24"],
["Massachusetts","6605058","25"],
["Michigan","9886095","26"],
["Minnesota","5347740","27"],
["Mississippi","2976872","28"],
["Missouri","6007182","29"],
["Montana","998554","30"],
["Nebraska","1841625","31"],
["Nevada","2730066","32"],
["New Hampshire","1319171","33"],
["New Jersey","8832406","34"],
["New Mexico","2069706","35"],
["New York","19487053","36"],
["North Carolina","9651380","37"],
["North Dakota","689781","38"],
["Ohio","11549590","39"],
["Oklahoma","3785742","40"],
["Oregon","3868721","41"],
["Pennsylvania","12731381","42"],
["Rhode Island","1051695","44"],
["South Carolina","4679602","45"],
["South Dakota","825198","46"],
["Tennessee","6402387","47"],
["Texas","25639373","48"],
["Utah","2813673","49"],
["Vermont","625904","50"],
["Virginia","8100653","51"],
["Washington","6819579","53"],
["West Virginia","1853619","54"],
["Wisconsin","5706871","55"],
["Wyoming","570134","56"],
["Puerto Rico","3682966","72"]]
```

The Hard Part – Data Mining HTML

- An enormous amount of things can go wrong.
  - The field ID may change in a website update.
  - Server may go down.
  - The field may not exist on some pages.
  - Corrupt HTML.
    - Yes, it happened. A website didn't close a tag properly.

- Tools Used
  - Python
  - Beautiful Soup (HTML parsing)
  - MongoDB
  - Amazon EC2 micro server

Funny Story

- I started data mining over 2 years ago.
- I use Amazon EC2 micro instance server (Ubuntu AMI 613 MB)

- My script stopped running when I exit SSH.
  - Nohup (no hangup)

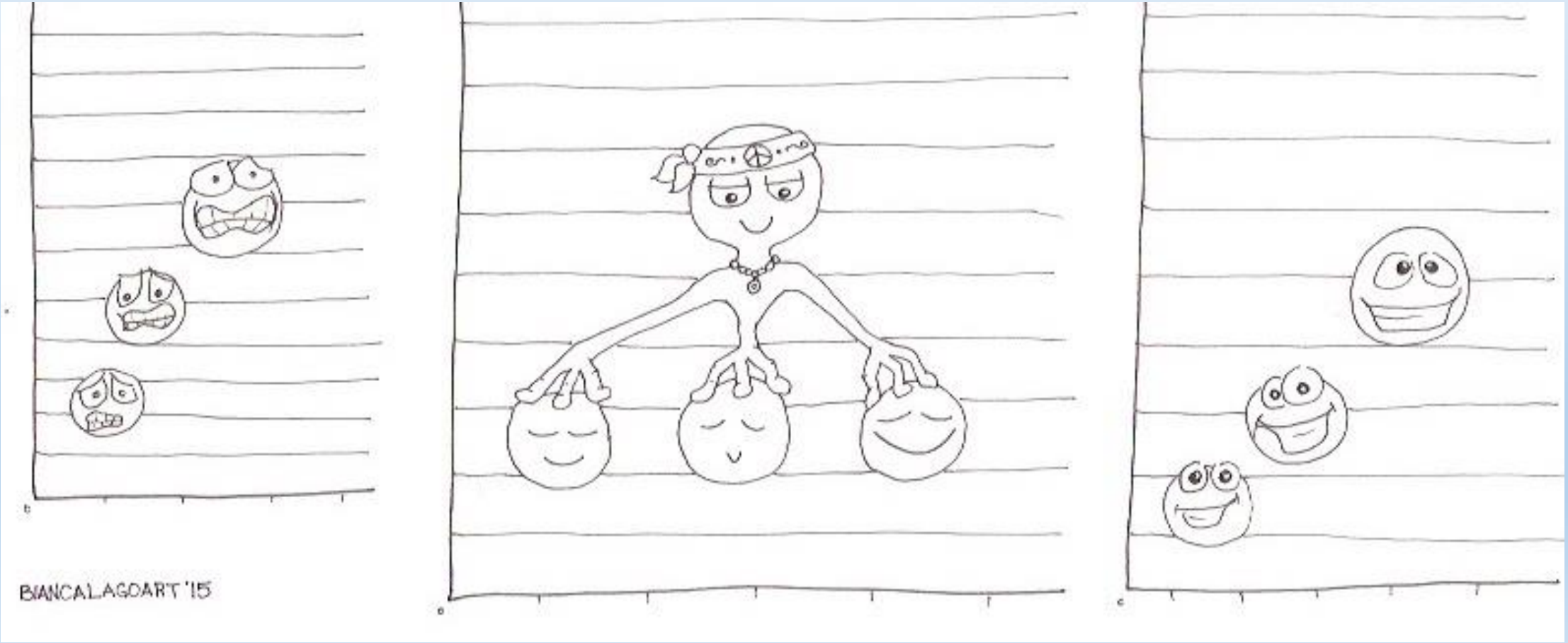
- My script stopped running but left no error logs.
  - Swapon (check if you have swap space setup)

U.S. Gazetteer Data

- What are the boundaries for every county in the U.S.?
  - The U.S. Gazetteer Office keeps track of the boundaries.
- The border of a county is defined by a set of coordinate pairs.
- Source: https://www.census.gov/geo/maps-data/data/gazetteer.html

Massaging Data

- Data are tense and need a good massage to reveal their secrets.



Massaging Data

- The U.S. Census Data are tidy.
  - Tidy but tedious.
- The HTML data are difficult to tidy up.
- The Gazetteer data require some finagling.

Massaging Data – U.S. Census Data

- U.S. Census data are binned by age group.
  - 18-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59, 60-64
- No “singles” variable.
  - Have to use never married, widowed, divorced, and married but separated variables.
  - Can’t do “all minus married” because there is no all variable in this dataset.
- Variables are available at a specific level e.g. by county, by state, etc

Massaging Data - HTML

- The bottleneck is the appropriate rate of making requests.
- Use Idle time to extract desired data from HTML.
- My script has been running for almost 2 years.
- About 6.8 million profiles were collected.

Massaging Data – The Big Problem

- Location is a user text input. (very scary)
  - Lots of misspelling and non-location text.
  - E.g. “Portland all the way”, “Miami babe”, “I live in Seattle”, “Oregano”, “Cali”
- Location data is “city, state” but I need the county and state.
  - What cities are in a county?
  - What county is Hillsboro in?
  - What county is Seattle in?
- Luckily the structure was always CITY_TEXT, STATE_TEXT

Massaging Data – Matching User Text

- Clean the text to remove frivolous terms.
- Levenshtein distance or Ratcliff/Obershelp to measure distance between two strings.
- There are over 150,000 cities and more than 6.8 million profiles.
- Time complexity problem.
- Match the state first and eliminate 1/51 possible city matches.
  - Congratulations D.C. I called you a state.
  - Eliminates multiple cities with the same name e.g. Kansas City in Missouri, Oregon, and Kansas.
- A few ad-hoc optimizations.

Massaging Data – Matching User Text

```
import Levenshtein as lev
"""Levenshtein distance computes the minimum number of edits needed to transform one string into the other"""
print(lev.ratio('Portland', 'Portlandia'))
```

```
0.888888888889
```

```
import difflib
"""(Ratcliff/Obershelp) Compute the similarity of two strings as the number of matching characters divided
by the total number of characters in the two strings. Matching characters are those in the longest common
subsequence plus, recursively, matching characters in the unmatched region on either side of the longest common
subsequence.
This does not yield minimal edit sequences, but does tend to yield matches that “look right” to people."""
print(difflib.SequenceMatcher(None, 'Portland', 'Portlandia').ratio())
```

```
0.888888888889
```

```
import us
print(difflib.get_close_matches('Oregano', [x.name for x in us.STATES], cutoff=0.3, n=3))
print(difflib.get_close_matches('Cali', [x.name for x in us.STATES], cutoff=0.3, n=3))
```

```
[u'Oregon', u'Virginia', u'Oklahoma']
[u'California', u'South Carolina', u'North Carolina']
```

Massaging Data – Matching User Text

- There are many profiles from populous cities.
- Why perform a computationally expensive operation again? Don’t.
- Store a result object in a hash table.
  - The lookup time is 1 – independent of the number of elements.
  - Check for a precomputed result before performing operations on location text.
- Overall ~94% of profiles retrieved their results from the hash table.
  - Dramatic improvement in processing time.
- Update the database with the state, county, and city FIPS number.
  - Glad I used MongoDB.

Massaging Data - Limitations

- The profile ID increments by one after a new profile is created.
- We can reasonably assume that the profiles from a random sample of these ID's will reflect the actual gender ratio.
- The ratio is biased for two main reasons:
  - Women are very likely to receive an uncomfortable message and delete their account shortly after. (which wouldn't get counted in the data)
  - Women are hesitant to create an account.
  - http://www.pewinternet.org/2013/10/21/online-dating-relationships/

Massaging Data – Making Assumptions

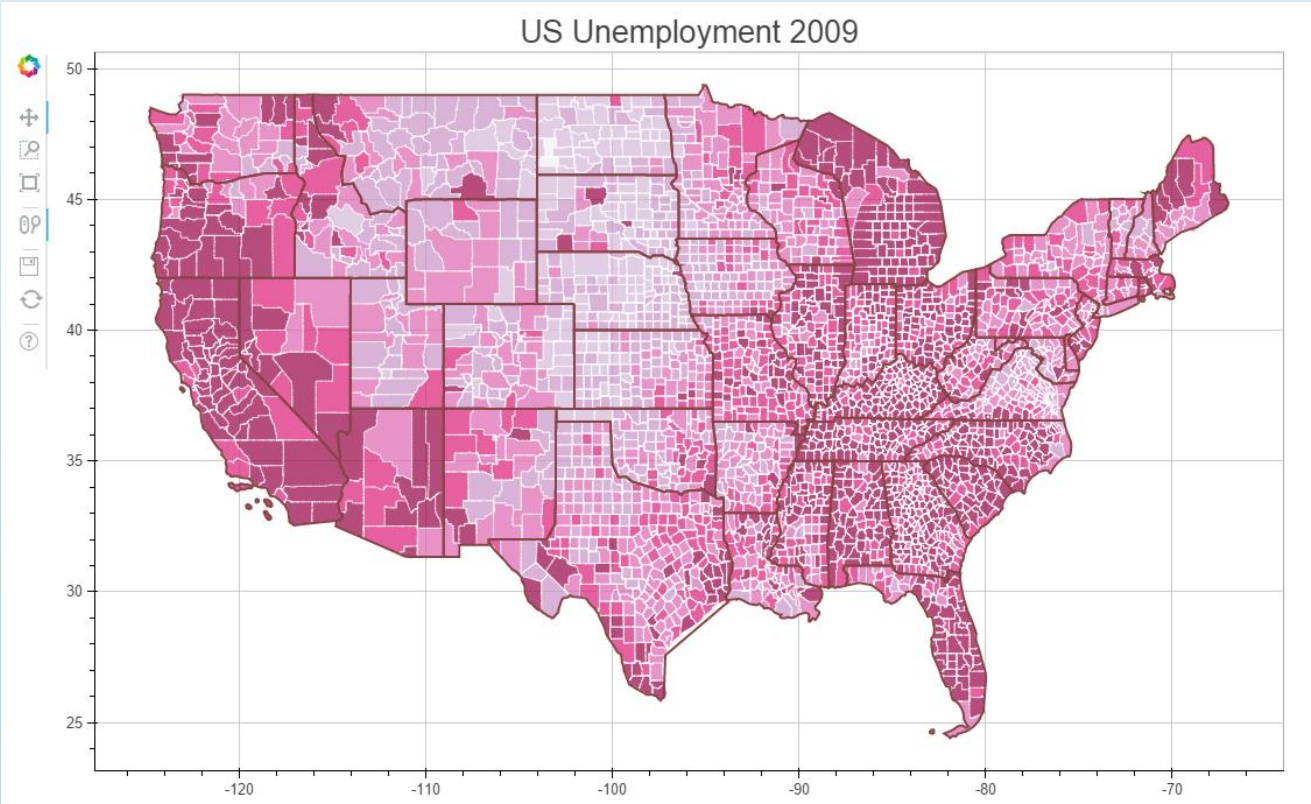
- There are too many assumptions required to interpret the ratio as absolute.
- Assume these distortions occur with equal probability in every county.
- The absolute ratio will be biased and useless but the difference in ratios between counties is useful.

Massaging Data - Gazetteer

- Need a database with state, county, city, fips, and boundary information.
- There is no official database with all these data. ☹
  - There is a state, county, and fips db.
  - There is a city and full fips db.
  - There is a fips and county boundary db.
- The solution is trivial but tedious.
- Also need to know what counties are in a metro area.

Visualizing Data

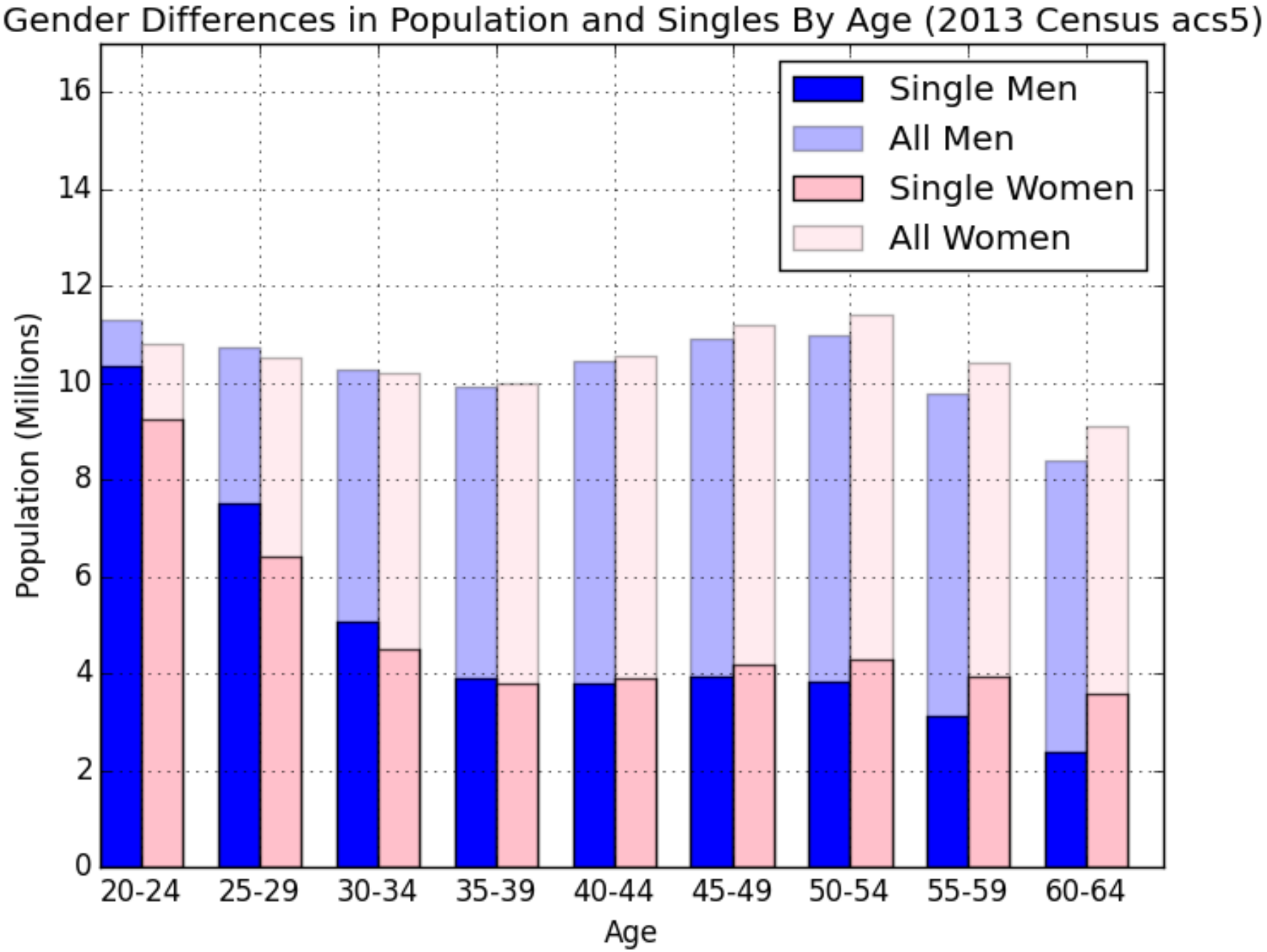
- Python Bokeh looks interesting.



http://bokeh.pydata.org/en/latest/docs/gallery/choropleth.html

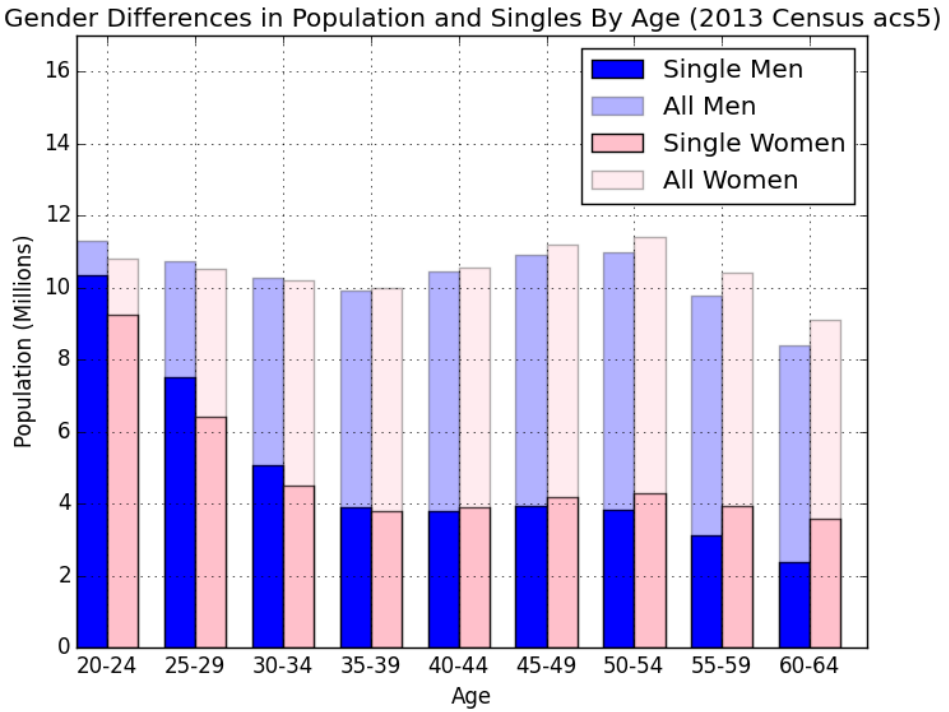
Visualizing Data

- Python Bokeh was rather slow for an extremely detailed map.
- There are many other options but none were free and set-and-forget.
- Well hello Google Fusion Tables.
  - For every zoom level there is a different level of granularity.
  - Very fast for a ~21MB file.
- Matplotlib


Gender Differences in Population and Singles By Age (2013 Census acs5)
Single Men
All Men
Single Women
All Women
Population (Millions)
0
2
4
6
8
10
12
14
16
20-24
25-29
30-34
35-39
40-44
45-49
50-54
55-59
60-64
Age

Discussing the Chart

- The age range in my project is 18-39.
- Most projects use a wider age range and report a comfortable 1:1 ratio.
- Men, generally, are in relationships with younger women.
- In the age range I selected there was not one metro area with a ratio as low as 1:1.



Age difference in heterosexual married couples, 2013 US Current Population Survey[4]

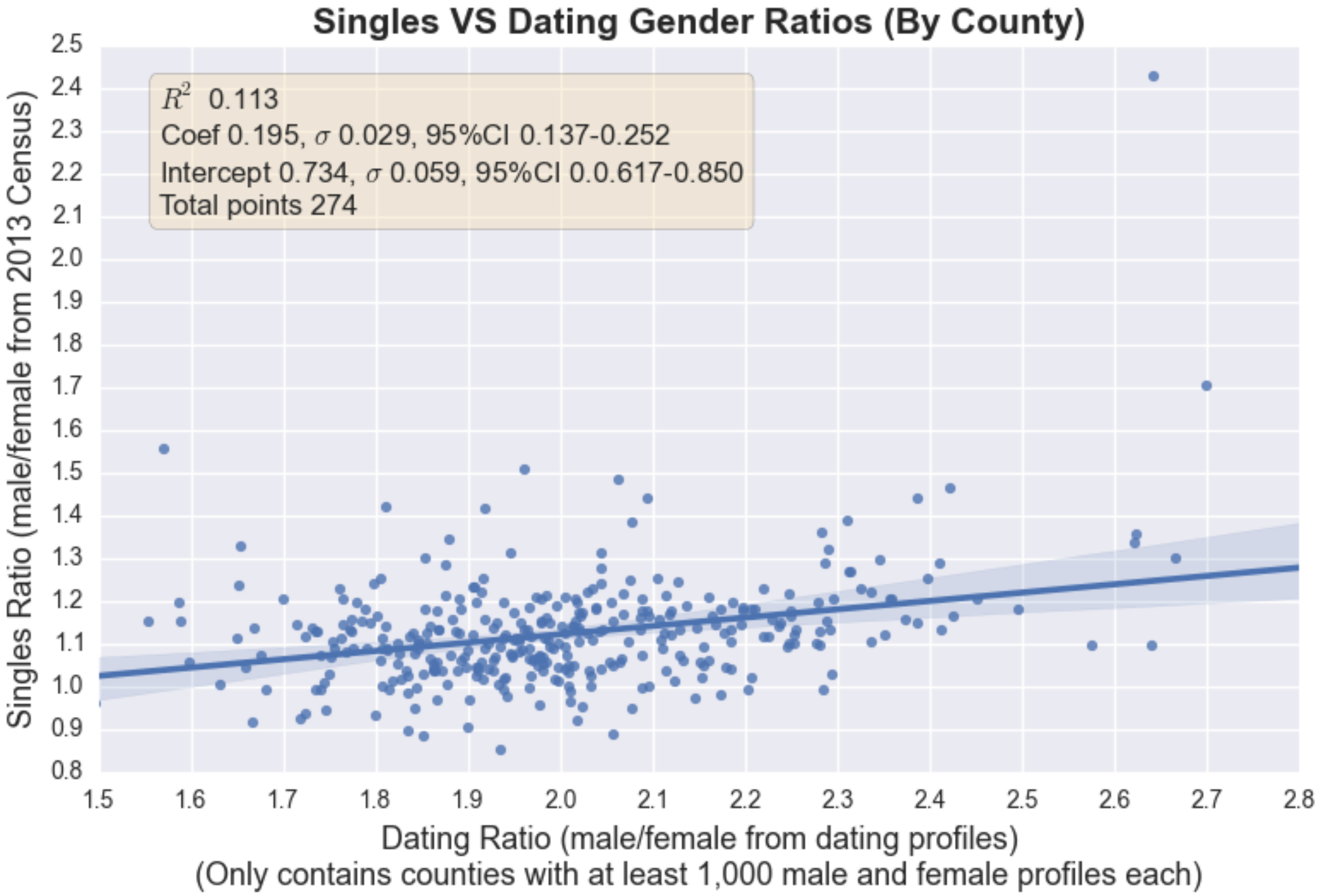
Age difference	Percentage of All Married Couples
Husband 20+ years older than wife	1.0
Husband 15–19 years older than wife	1.6
Husband 10–14 years older than wife	4.8
Husband 6–9 years older than wife	11.6
Husband 4–5 years older than wife	13.3
Husband 2–3 years older than wife	20.4
Husband and wife within 1 year	33.2
Wife 2–3 years older than husband	6.5
Wife 4–5 years older than husband	3.3
Wife 6–9 years older than husband	2.7
Wife 10–14 years older than husband	1.0
Wife 15–19 years older than husband	0.3
Wife 20+ years older than husband	0.3

https://en.wikipedia.org/wiki/Age_disparity_in_sexual_relationships

Singles VS Dating Gender Ratios

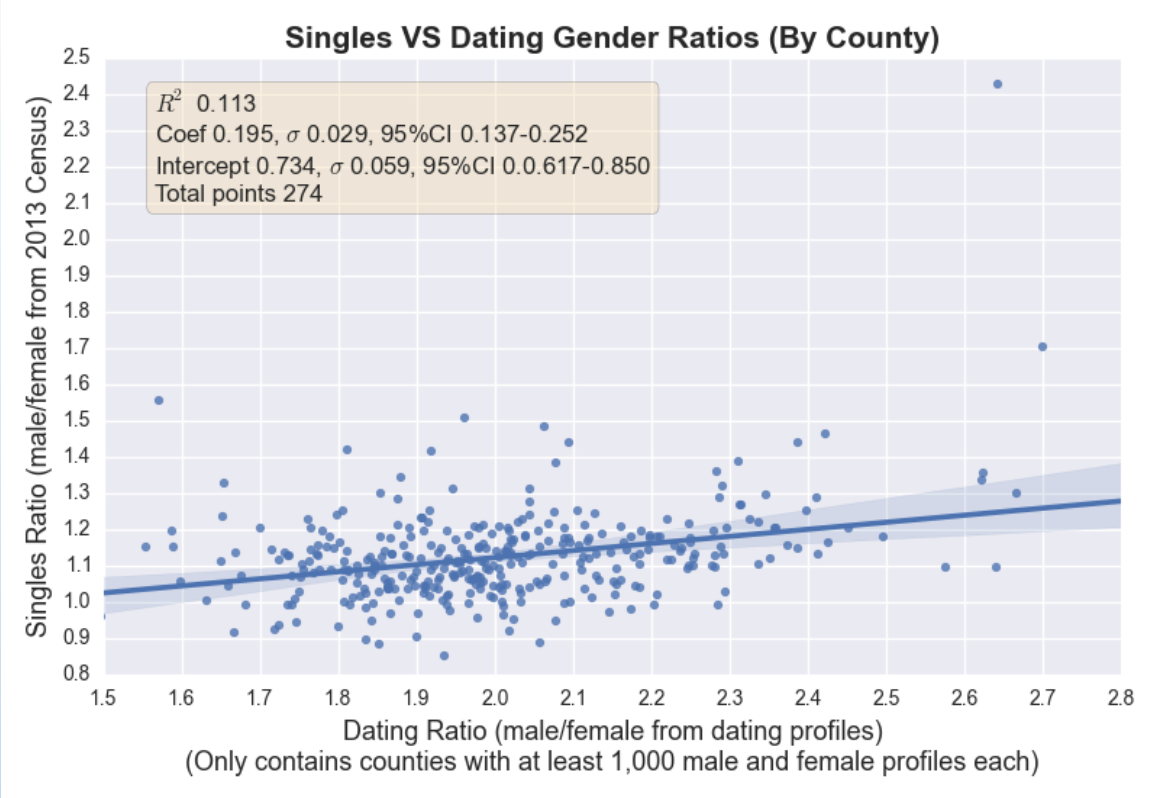
- Dating Ratio: The gender ratio from public dating profiles. (male profiles) / (female profiles)
- Singles Ratio: The gender ratio of single people from the 2013 Census. (male/female)

- Is there a relationship between these ratios?
- One would expect a county with more single men to also have a higher dating ratio.


Singles VS Dating Gender Ratios (By County)
R^2 0.113
Coef 0.195, σ 0.029, 95%CI 0.137-0.252
Intercept 0.734, σ 0.059, 95%CI 0.0.617-0.850
Total points 274
Singles Ratio (male/female from 2013 Census)
2.5
2.4
2.3
2.2
2.1
2.0
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0.9
0.8
1.5
1.6
1.7
1.8
1.9
2.0
2.1
2.2
2.3
2.4
2.5
2.6
2.7
2.8
Dating Ratio (male/female from dating profiles)
(Only contains counties with at least 1,000 male and female profiles each)

Discussing the Chart

- Coefficient zero is not included in a 95% confidence interval.
- A strong R-Squared would mean that too much variability in the dating ratio is explained by the singles ratio.
  - Then the dating ratio would provide little extra information.

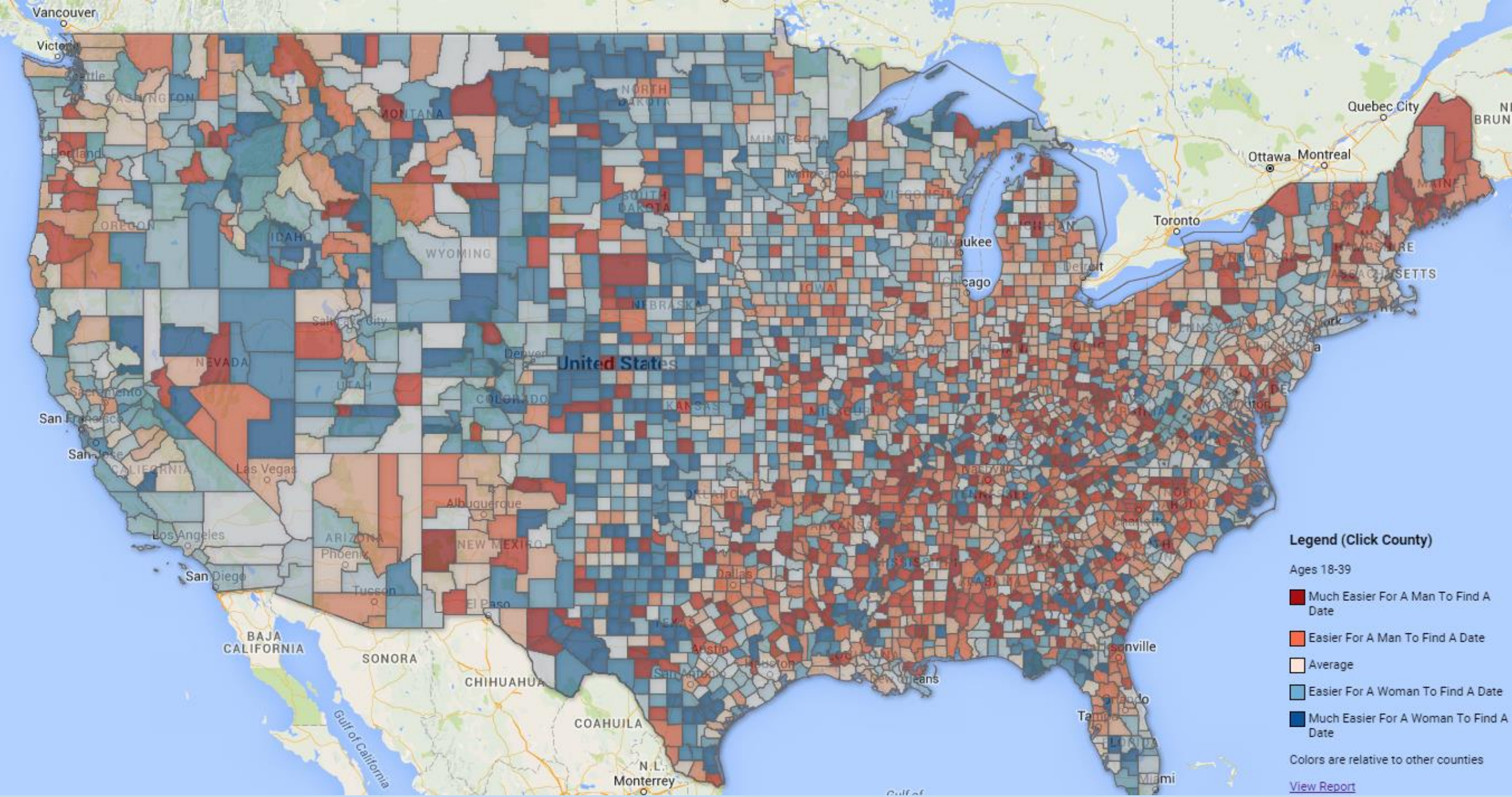


May social dynamics can't be captured by marriage rate. e.g. For a city with a singles ratio of 1.1 and a dating ratio of 2.6 there may be many people in relationships w/o being married.

Thought experiment: Imagine there are 100 single women and 110 single men. The ratio is 1.1. Now imagine 90 of them are in a relationship. Then the ratio becomes 2.

Bringing All Parts Of The Project Together

- Dating Ratio: The gender ratio from public dating profiles. (total male profiles) / (total female profiles)
- Singles Ratio: The gender ratio of single people from the 2013 Census. (male/female)
- Combined Ratio: SQRT(Dating Ratio * Singles Ratio)
- Show the map.



http://thomaz.me/dating_map.html

Pew Research Center

http://www.pewresearch.org/fact-tank/2014/10/02/the-best-and-worst-cities-for-women-looking-to-marry/

Results - The Easiest and Hardest Metro Areas to Find a Date

List of best places to find a date.
For metro areas with over 1.5 million people.
For all races in the 18-39 age range.
Dating Ratio: The gender ratio from online dating profiles. (total male profiles) / (total female profiles)
Singles Ratio: The gender ratio of single people from the 2010 Census. (male/female)
Combined Ratio: SQRT(Dating Ratio * Singles Ratio)
Rank 1 = Easiest place for a woman to find a date (hardest for a man)
Rank 39 = Hardest place for a woman to find a date (easiest for a man)

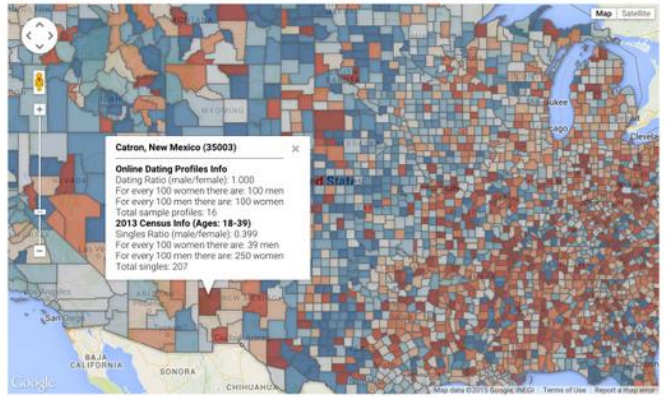
Show 10 entries Search:

Rank	Metro Area	Combined Ratio	Dating Ratio	Singles Ratio
1	San Jose-Sunnyvale-Santa Clara, CA	1.852	2.649	1.296
2	San Diego-Carlsbad, CA	1.714	2.313	1.270
3	San Francisco-Oakland-Hayward, CA	1.676	2.478	1.133
4	Denver-Aurora-Lakewood, CO	1.637	2.291	1.170
5	Los Angeles-Long Beach-Anaheim, CA	1.617	2.280	1.147
6	Miami-Fort Lauderdale-West Palm Beach, FL	1.600	2.275	1.125
7	Riverside-San Bernardino-Ontario, CA	1.578	2.146	1.161
8	Virginia Beach-Norfolk-Newport News, VA-NC	1.563	2.051	1.191
9	Houston-The Woodlands-Sugar Land, TX	1.552	2.087	1.155
10	Austin-Round Rock, TX	1.550	2.074	1.158

Rank	Metro Area	Combined Ratio	Dating Ratio	Singles Ratio
39	Charlotte-Concord-Gastonia, NC-SC	1.376	1.794	1.055
38	St. Louis, MO-IL	1.382	1.830	1.044
37	Atlanta-Sandy Springs-Roswell, GA	1.397	1.854	1.053
36	Baltimore-Columbia-Towson, MD	1.404	1.920	1.027
35	Nashville-Davidson--Murfreesboro--Franklin, TN	1.409	1.864	1.065
34	Indianapolis-Carmel-Anderson, IN	1.413	1.874	1.065
33	Kansas City, MO-KS	1.421	1.855	1.089
32	Dallas-Fort Worth-Arlington, TX	1.431	1.846	1.109
31	Tampa-St. Petersburg-Clearwater, FL	1.439	1.942	1.067
30	Philadelphia-Camden-Wilmington, PA-NJ-DE-MD	1.444	1.996	1.044

Featured on FiveThirtyEight.com

1.7 men for every woman: Reddit user Thomaz Santana wanted to find the easiest and hardest places to find a date. He calculated the ratio of single men to single women in each U.S. county using census data and created a similar dating ratio with data from Plenty of Fish, an online dating site. Both sources are somewhat problematic: The resulting singles ratio is hard to interpret because it fails to take into account sexual orientation, while the dating ratio ignores the fact that men and women aren't equally likely to create an account. Still, both ratios are interesting. According to Santana's analysis, the place where it's easiest for a woman to find a male date is the San Jose, California, metro area, where single men outnumber women 1.3 to 1 and where there are 2.6 male dating profiles for every female profile. The place where it's hardest for a woman to find a date with a man is Atlanta. [Thomaz Lago Santana]



http://fivethirtyeight.com/datalab/the-week-in-data-bloggers-in-basements-sunny-spots-and-jell-o/

- Really cool visualization of census data but no tutorial on how to do it.
- I created a tutorial and uploaded the source code and the gazetteer db on github.
- Tutorial got featured on Python Weekly newsletter.

Q&A Time.