Economic_Connectedness_Assignment

April 9, 2023

1 Economic Connectedness

In this assignment you will replicate partly two studies carried out on social capital. The studies appeared in the journal Nature:

- Chetty, R., Jackson, M.O., Kuchler, T. et al. Social capital I: measurement and associations with economic mobility. Nature 608, 108–121 (2022). https://doi.org/10.1038/s41586-022-04996-4.
- Chetty, R., Jackson, M.O., Kuchler, T. et al. Social capital II: determinants of economic connectedness. Nature 608, 122–134 (2022). https://doi.org/10.1038/s41586-022-04997-3.

Read the papers, locate, download, and familiarize yourself with the dataset provided by the authors.

Panos Louridas, Associate Professor Department of Management Science and Technology Athens University of Economics and Business louridas@aueb.gr

1.1 Questions

1.1.1 Q1: The Geography of Social Capital in the United States

You will replicate Figure 2a of the first paper. But while the figure in the paper is static, you will create an interactive figure, like the one that is available online at https://www.socialcapital.org/. You can see an example of what you should do here.

In the figure, the social capital is represented by the Economic Connectedness (EC). Economic Connectedness is the degree to which people with low and high Socioeconomic Status (SES) are friends with each other. More formally, to define EC we start by measuring each individual's i share of friends from SES quantile Q:

$$f_{Q,i} = \frac{[\text{Number of friends in SES quantile } Q]_i}{\text{Total number of friends of } i}$$

Then we normalize $f_{Q,i}$ by the share of individuals in the sample who belong to quantile Q, w_Q (for example, $w_Q = 0.1$ for deciles) to get the Individual Economic Connectedness (IEC):

$$\text{IEC}_{Q,i} = \frac{f_{Q,i}}{w_Q}$$

The level of EC in a community c is the defined as the mean level of individual EC of low-SES L (for example, below-median) members of that community, as follows:

$$\mathrm{EC}_c = \frac{\sum_{i \in L \cap c} \mathrm{IEC}_i}{N_{Lc}}$$

where N_{Lc} is the number of low-SES individuals in community c.

In the figure and in what follows, the EC is twice the share of friends with above-median SES among people with below-median SES; that follows from the above definitions for $w_Q = 0.5$.

The map is constructed using Plotly. The data are displayed per county. When the pointed hovers each county, it should display the name of the county and the state it belongs to, the code of the county, and the Economic Connectedness of the county. If there are no data for a particular county, it should be painted with a distinct color (in our example it is painted gold) and the economic connectedness should be given as "NA".

Data for social capital can be found at the Social Capital Atlas Datasets.

1.1.2 Q2: Economic Connectedness and Outcomes

You will replicate Figure 4 of the first paper. The figure is a scatter plot of upward income mobility against economic connectedness (EC) for the 200 most populous US counties. The income mobility is obtained from the Opportunity Atlas, whose replication data can be found here. Your figure should look like the following one.

1.1.3 Q3: Upward Income Mobility, Economic Connectedness, and Median House Income

You will replicate Figure 6 of the first paper. The figure is a scatter plot of economic connectedness (EC) against median household income. You will need to compile data from replication package of the papers with data from the Social Capital Atlas Datasets. The color of the dots corresponds to the child's income rank in adulthood given that the parents' income is in the 25th percentile. The colors correspond to five intervals, which are the quintiles dividing our data. Your figure should look like the following one.

1.1.4 Q4: Friending Bias and Exposure by High School

You will replicate Figure 5a of the second paper. The figure depicts the Socioeconomic Status (SES) of parents against the friending bias of students of low SES, with data from the Social Capital Atlas Datasets.

Note that to get the share of high-parental-SES students, which is the x-axis, you need to take the economic connectedness with parental SES and divide it by two. That is because the economic connectedness with parental SES is defined as two times the share of high-parental-SES friends among low-parental-SES individuals, averaged over all low-parental-SES individuals at the school.

Note also that both x and y axis are percentages and the y axis is reversed.

In the end, your figure should look like the one below. Text placement might differ slightly. The annoted high schools are `00941729`, `060474000432`, `170993000942`, `170993001185`, `170993003989`, 171449001804, 250327000436, 360009101928, 370297001285, 483702004138, 250843001336, 062271003230, 010237000962, 00846981, 00852124.

1.1.5 Q5: Friending Bias vs. Racial Diversity

You will replicate Extended Data Figure 3 of the second paper. The figure depicts friending bias against racial diversity. Racial diversity is defined by the Herfindahl-Hirschman Index (HHI), borrowed from investing. Translated here, it is \$ 1− _{i}{s_i}^2\$, where s_i is the fraction of race/ethnicity i (Black, White, Asian, Hispanic, Native American).

As you can see, the figure contains two scatter plots with their respective regression lines, one for college data and the other for neighborhood data. Each of the two plots displays binned data (that's why you don't see loads of dots and diamonds). The bins are produced by dividing the x-axis into ventiles (i.e., 5 percentile point bins); then we plot the mean of the y-axis variable against the appropriate mean of the x-axis variable in each ventile.

The mean of the x-axis variable, the HHI index, is the weighted mean of HHI:

- For the college plot, the weights are given by the mean number of students per cohort.
- For the neighborhood plot, the weights are given by the number of children with below-national-median parental household income.

The y-axis variable:

- For the college plot, it is the mean of the college friending bias.
- For the neighborhood plot, it is the mean of the neighborhood friending bias.

In the end, your figure should look like the following.

1.2 Beginning of Assignment

1.2.1 Important note: the datasets are all uploaded here https://drive.google.com/file/d/1gB5u21dc_X7oZoyBR7W5MzkoVXFat7jP/view?usp=sha

1.3 Q1: The Geography of Social Capital in the United States

First of all, we download the datasets from https://www.socialcapital.org/ and place them in a folder that we name dataset.

After that, we need to import all needed libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

We then load the county dataset onto a dataframe

[2]:
```
county = pd.read_csv('dataset/social_capital_county.csv')
county
```

[2]:

```
      county         county_name  num_below_p50   pop2018  ec_county  \
0       1001     Autauga, Alabama     5922.39210   55200.0    0.72077
1       1003     Baldwin, Alabama    15458.39600  208107.0    0.74313
2       1005     Barbour, Alabama     4863.97360   25782.0    0.41366
3       1007        Bibb, Alabama     3061.49340   22527.0    0.63152
4       1009      Blount, Alabama     6740.91160   57645.0    0.72562
…        …                   …              …       …         …
3084   56037  Sweetwater, Wyoming     2402.96900   44117.0    0.96235
3085   56039       Teton, Wyoming      783.24982   23059.0    1.07623
3086   56041       Uinta, Wyoming     2174.06180   20609.0    0.95452
3087   56043    Washakie, Wyoming      872.51544    8129.0    0.90667
3088   56045      Weston, Wyoming      635.28436    7100.0    0.97840

      ec_se_county  child_ec_county  child_ec_se_county  ec_grp_mem_county  \
0          0.00831          1.11754             0.02467            0.77223
1          0.00661          0.83064             0.01629            0.76215
2          0.00978          0.58541             0.02707            0.35927
3          0.01175          0.72265             0.03027            0.68094
4          0.00985          0.76096             0.02466            0.79584
…              …                …                  …                 …
3084       0.01280          1.14781             0.02794            1.13449
3085       0.01744          1.23113             0.04692            1.13296
3086       0.01404          1.04595             0.03455            0.92831
3087       0.01928          0.90794             0.04962            0.78223
3088       0.02036          1.09118             0.05823            0.93135

      ec_high_county  …  child_exposure_county  child_high_exposure_county  \
0            1.21372  …                1.14816                     1.19944
1            1.28302  …                0.84588                     1.00797
2            0.91897  …                0.63306                     0.71967
3            1.06378  …                0.71433                     0.72395
4            1.10569  …                0.74821                     0.79375
…                …  …                    …                          …
3084         1.32399  …                1.12164                     1.12907
3085         1.63551  …                1.32874                     1.35341
3086         1.32040  …                1.05446                     1.06284
3087         1.29208  …                0.88480                     0.88589
3088         1.28553  …                1.03325                     1.05526

      bias_grp_mem_county  bias_grp_mem_high_county  child_bias_county  \
0                 0.05526                  -0.22748            0.02668
1                 0.02950                  -0.21519            0.01802
2                 0.13457                  -0.34086            0.07528
3                 0.04108                  -0.27727           -0.01165
4                 0.00217                  -0.24946           -0.01704
…                     …                        …                 …
3084              0.09519                  -0.12030           -0.02333
```

```
3085              0.14337                    -0.11958           0.07346
3086              0.13816                    -0.12194           0.00808
3087              0.06667                    -0.20435          -0.02615
3088              0.02279                    -0.17229          -0.05606

      child_high_bias_county  clustering_county  support_ratio_county  \
0                   -0.08229            0.10347               0.98275
1                   -0.05241            0.09624               0.98684
2                   -0.19714            0.14911               0.99911
3                   -0.15993            0.14252               0.99716
4                   -0.08745            0.11243               0.99069
…                        …                 …                    …
3084                -0.08683            0.10809               0.99710
3085                -0.07364            0.09253               0.98648
3086                -0.06074            0.11204               0.99479
3087                -0.06076            0.11592               0.99708
3088                -0.04609            0.11927               0.99730

      volunteering_rate_county  civic_organizations_county
0                      0.04355                     0.01518
1                      0.06117                     0.01526
2                      0.02093                     0.01474
3                      0.05294                     0.01439
4                      0.05704                     0.01724
…                          …                          …
3084                   0.07321                     0.01225
3085                   0.09747                     0.03223
3086                   0.06942                     0.01222
3087                   0.05843                     0.03512
3088                   0.13635                     0.02375

[3089 rows x 26 columns]
```

If we try to visualise the data that we currently have, we notice that some counties are completely missing from the dataset, so we download this csv file which contains FIPS (county codes) and county names from this link: https://github.com/kjhealy/fips-codes/blob/master/county_fips_master.csv

note: when i ran the notebook on some pcs this needed an extra 'encoding = latin1' argument, i am not sure as to what caused this so i did not include it in the source code but just in case this doesn't run properly, thats the solution

[3]:
```
countyfips = pd.read_csv('dataset/county_fips_master.csv')
```

First of all we need to drop all useless columns from the dataframe

[4]:
```
countyfips = countyfips.drop(columns = ['state_abbr', 'long_name', 'sumlev',␣
↪'region', 'division', 'state', 'county', 'crosswalk', 'region_name',␣
↪'division_name'])
```

then we need to remove " County" from the column countyname, then create a new column and add the county's name and the state's name so it matches the original dataframe, andthen drop the now useless columns that we combined

[5]:
```
countyfips['county_name'] = countyfips['county_name'].apply(lambda x: x[:
 ↪len(x)-7])
countyfips['full_county_name'] = countyfips.apply(lambda row: row.county_name +␣
 ↪", " + row.state_name, axis=1)
countyfips = countyfips.drop(columns = ['county_name', 'state_name'])
```

this is what the secondary dataframe looks like just before merging

[6]:
```
countyfips
```

[6]:
```
       fips     full_county_name
0      1001     Autauga, Alabama
1      1003     Baldwin, Alabama
2      1005     Barbour, Alabama
3      1007        Bibb, Alabama
4      1009      Blount, Alabama
…      …                     …
3141  56037  Sweetwater, Wyoming
3142  56039       Teton, Wyoming
3143  56041       Uinta, Wyoming
3144  56043    Washakie, Wyoming
3145  56045      Weston, Wyoming

[3146 rows x 2 columns]
```

we merge on both columns and we also convert county codes to strings so we can play around with them a bit later

[7]:
```
county['county'] = county['county'].map(str)
countyfips['fips'] = countyfips['fips'].map(str)
county = county.merge(countyfips, how="outer", left_on="county",␣
 ↪right_on="fips")
```

we then take care of what's left of merging by combining the fips and county columns as well as the county name and full county name columns

[8]:
```
county.county = np.where(county.county.isnull(),
                              county.fips,
                              county.county)

county.county_name = np.where(county.county_name.isnull(),
                                   county.full_county_name,
                                   county.county_name)

county.drop(["fips", "full_county_name"], axis=1, inplace=True)
```

we want to replace NaN values on ec_county with 0 so we can visualise them with plotly and we change the data type of the county column to be a str.

important note, this can be dangerous if we mishandle the data since economic connectedness is an arithmetic value that can reach the value of zero so we need to be very careful with how we use the dataframe now since we cannot use functions like (avg) and (min) anymore, the data is made this way ONLY for visualisation purposes and these values need to be returned to NaN if we are to use the data for anything else

[9]:
```
county['ec_county'] = county['ec_county'].fillna(0)
```

We notice that the county collumn has 4 digit numbers in it, and in order to visualise county data with plotly the codes need to be 5 digits, so we add a 0 in front of every 4-digit entry

[10]:
```
county['county'] = county['county'].apply(lambda x: '0' + x if len(x)==4 else x)
```

finally we can actually visualise the data using plotly

[11]:
```
from urllib.request import urlopen
import json

colorscale=[[0, 'gray'],[0.1, 'gray'], [0.1, 'blue'],
            [0.6, "white"], [0.9, 'red'], [1, 'brown']]

with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/
↪geojson-counties-fips.json') as response:
    counties = json.load(response)

import plotly.express as px

fig = px.choropleth(county, geojson=counties, locations='county',␣
↪color='ec_county', color_continuous_scale=colorscale,
                           range_color=(county['ec_county'].min(),␣
↪county['ec_county'].max()),
                           hover_name="county_name", scope="usa",␣
↪labels={'ec_county':'Economic Connectedness'}
                          )
fig.update_layout(margin={"r":0,"t":0,"l":0,"b":0})
fig.show()
```

1.4 Q2: Economic Connectedness and Outcomes

First of all we need to import our data, we use the per-county data from the original dataset as well as the county_outcomes.csv file from the opportunity atlas dataset

[12]:
```
counties = pd.read_csv('dataset/social_capital_county.csv')
countykir = pd.read_csv('dataset/county_outcomes.csv')
```

```
c:\users\kharnifex\appdata\local\programs\python\python38\lib\site-
packages\IPython\core\interactiveshell.py:3012: DtypeWarning:
```

```
Columns (7886) have mixed types.Specify dtype option on import or set
low_memory=False.
```

[13]:
```
counties
```

[13]:
```
      county          county_name  num_below_p50   pop2018  ec_county  \
0       1001      Autauga, Alabama     5922.39210   55200.0    0.72077
1       1003      Baldwin, Alabama    15458.39600  208107.0    0.74313
2       1005      Barbour, Alabama     4863.97360   25782.0    0.41366
3       1007         Bibb, Alabama     3061.49340   22527.0    0.63152
4       1009       Blount, Alabama     6740.91160   57645.0    0.72562
…          …                     …              …        …          …
3084   56037  Sweetwater, Wyoming     2402.96900   44117.0    0.96235
3085   56039       Teton, Wyoming      783.24982   23059.0    1.07623
3086   56041       Uinta, Wyoming     2174.06180   20609.0    0.95452
3087   56043    Washakie, Wyoming      872.51544    8129.0    0.90667
3088   56045      Weston, Wyoming      635.28436    7100.0    0.97840

      ec_se_county  child_ec_county  child_ec_se_county  ec_grp_mem_county  \
0          0.00831          1.11754             0.02467            0.77223
1          0.00661          0.83064             0.01629            0.76215
2          0.00978          0.58541             0.02707            0.35927
3          0.01175          0.72265             0.03027            0.68094
4          0.00985          0.76096             0.02466            0.79584
…                …                …                   …                  …
3084       0.01280          1.14781             0.02794            1.13449
3085       0.01744          1.23113             0.04692            1.13296
3086       0.01404          1.04595             0.03455            0.92831
3087       0.01928          0.90794             0.04962            0.78223
3088       0.02036          1.09118             0.05823            0.93135

      ec_high_county  …  child_exposure_county  child_high_exposure_county  \
0            1.21372  …                1.14816                     1.19944
1            1.28302  …                0.84588                     1.00797
2            0.91897  …                0.63306                     0.71967
3            1.06378  …                0.71433                     0.72395
4            1.10569  …                0.74821                     0.79375
…                  …  …                      …                           …
3084         1.32399  …                1.12164                     1.12907
3085         1.63551  …                1.32874                     1.35341
3086         1.32040  …                1.05446                     1.06284
3087         1.29208  …                0.88480                     0.88589
3088         1.28553  …                1.03325                     1.05526

      bias_grp_mem_county  bias_grp_mem_high_county  child_bias_county  \
```

```
0                 0.05526                    -0.22748             0.02668
1                 0.02950                    -0.21519             0.01802
2                 0.13457                    -0.34086             0.07528
3                 0.04108                    -0.27727            -0.01165
4                 0.00217                    -0.24946            -0.01704
…                   …                           …                  …
3084              0.09519                    -0.12030            -0.02333
3085              0.14337                    -0.11958             0.07346
3086              0.13816                    -0.12194             0.00808
3087              0.06667                    -0.20435            -0.02615
3088              0.02279                    -0.17229            -0.05606

      child_high_bias_county  clustering_county  support_ratio_county  \
0                   -0.08229            0.10347               0.98275
1                   -0.05241            0.09624               0.98684
2                   -0.19714            0.14911               0.99911
3                   -0.15993            0.14252               0.99716
4                   -0.08745            0.11243               0.99069
…                      …                  …                    …
3084                -0.08683            0.10809               0.99710
3085                -0.07364            0.09253               0.98648
3086                -0.06074            0.11204               0.99479
3087                -0.06076            0.11592               0.99708
3088                -0.04609            0.11927               0.99730

      volunteering_rate_county  civic_organizations_county
0                      0.04355                     0.01518
1                      0.06117                     0.01526
2                      0.02093                     0.01474
3                      0.05294                     0.01439
4                      0.05704                     0.01724
…                        …                           …
3084                   0.07321                     0.01225
3085                   0.09747                     0.03223
3086                   0.06942                     0.01222
3087                   0.05843                     0.03512
3088                   0.13635                     0.02375

[3089 rows x 26 columns]
```

[14]:
```
countykir
```

[14]:
```
      state  county  kir_natam_female_p1  kir_natam_female_p25  \
0         1       1                  NaN                   NaN
1         1       3               0.3436              0.343627
2         1       5                  NaN                   NaN
3         1       7                  NaN                   NaN
```

```
4         1       9                   NaN                   NaN
…     …     …                   …                    …
3214     72     145                   NaN                   NaN
3215     72     147                   NaN                   NaN
3216     72     149                   NaN                   NaN
3217     72     151                   NaN                   NaN
3218     72     153                   NaN                   NaN

      kir_natam_female_p50  kir_natam_female_p75  kir_natam_female_p100  \
0                      NaN                   NaN                    NaN
1                 0.343645              0.343667               0.343722
2                      NaN                   NaN                    NaN
3                      NaN                   NaN                    NaN
4                      NaN                   NaN                    NaN
…                    …                     …                      …
3214                   NaN                   NaN                    NaN
3215                   NaN                   NaN                    NaN
3216                   NaN                   NaN                    NaN
3217                   NaN                   NaN                    NaN
3218                   NaN                   NaN                    NaN

      kir_natam_female_n  kir_natam_female_mean  jail_natam_female_p1  …  \
0                    NaN                    NaN                   NaN  …
1                   42.0               0.341199             -0.010921  …
2                    NaN                    NaN                   NaN  …
3                    NaN                    NaN                   NaN  …
4                    NaN                    NaN                   NaN  …
…                  …                     …                     …   …
3214                 NaN                    NaN                   NaN  …
3215                 NaN                    NaN                   NaN  …
3216                 NaN                    NaN                   NaN  …
3217                 NaN                    NaN                   NaN  …
3218                 NaN                    NaN                   NaN  …

      coll_white_pooled_mean_se  comcoll_white_pooled_mean_se  \
0                      0.020800                      0.021270
1                      0.014500                      0.014719
2                      0.035349                      0.038319
3                      0.040235                      0.040666
4                      0.018691                      0.022029
…                          …                            …
3214                        NaN                           NaN
3215                        NaN                           NaN
3216                        NaN                           NaN
3217                        NaN                           NaN
3218                        NaN                           NaN
```

	somecoll_white_pooled_mean_se	hs_white_pooled_mean_se	\
0	0.020339	0.012137	
1	0.012726	0.007792	
2	0.030695	0.019642	
3	0.043610	0.025271	
4	0.020685	0.012227	
…	…	…	
3214	NaN	NaN	
3215	NaN	NaN	
3216	NaN	NaN	
3217	NaN	NaN	
3218	NaN	NaN	

	wgflx_rk_white_pooled_mean_se	hours_wk_white_pooled_mean_se	\
0	0.018251	1.131328	
1	0.012266	0.741782	
2	0.027699	1.673795	
3	0.036064	2.380808	
4	0.017313	1.158366	
…	…	…	
3214	NaN	NaN	
3215	NaN	NaN	
3216	NaN	NaN	
3217	NaN	NaN	
3218	NaN	NaN	

	kfr_native_white_pooled_mean_se	kir_native_white_pooled_mean_se	\
0	0.008103	0.008534	
1	0.005500	0.005603	
2	0.013528	0.013531	
3	0.016382	0.016979	
4	0.007895	0.008158	
…	…	…	
3214	NaN	NaN	
3215	NaN	NaN	
3216	NaN	NaN	
3217	NaN	NaN	
3218	NaN	NaN	

	kir_imm_white_pooled_mean_se	kfr_imm_white_pooled_mean_se
0	0.057445	0.058009
1	0.041219	0.037302
2	NaN	NaN
3	NaN	NaN
4	NaN	NaN
…	…	…
3214	NaN	NaN

```
3215                              NaN                            NaN
3216                              NaN                            NaN
3217                              NaN                            NaN
3218                              NaN                            NaN

[3219 rows x 10827 columns]
```

We use the city and county codes on the opportunity atlas dataframe to generate a fips column so we can merge the two dataframes later

```
[15]: countykir['fips'] = countykir.apply(lambda x: x.state*1000 + x.county , axis=1)
      countykir['fips'] = countykir['fips'].map(str)
```

we sort the original dataframe by population and create a new dataframe which includes only the top 200 entries

```
[16]: counties = counties.sort_values('pop2018', ascending=False)
      counties2h = counties.head(200).copy()
```

After looking through the documentation of the almost 11 thousand columns of the dataframe we got off opportunity atlas we come to the realization that the way the data is stored in that file is through the worst implementation of a data cube ever known to man with the 3 dimensions being race, gender and percentile and we want the (null),(null),p25 value which for some reason is labeled as pooled_pooled_p25

So we rename the column into something a human can understand and we create a new dataframe that only includes said column and the fips code

```
[17]: countykir = countykir.rename(columns={'kir_pooled_pooled_p25':␣
       ↪'income_rank_p25'})
      countykir2 = countykir.loc[:,['income_rank_p25', 'fips']]
      countykir2
```

```
[17]:       income_rank_p25   fips
      0            0.384716   1001
      1            0.407555   1003
      2            0.397180   1005
      3            0.380409   1007
      4            0.383874   1009
      …                 …      …
      3214         0.367308  72145
      3215              NaN  72147
      3216         0.286394  72149
      3217         0.392320  72151
      3218         0.405954  72153

      [3219 rows x 2 columns]
```

We then narrow down the amount of columns on the dataframe with the top 200 counties before merging so that the merge is more memory-efficient

[18]:
```
counties2h = counties2h.loc[:,['county', 'county_name','pop2018', 'ec_county']]
counties2h
```

[18]:
```
      county              county_name     pop2018  ec_county
203     6037  Los Angeles, California  10098052.0    0.73580
605    17031           Cook, Illinois   5223719.0    0.75869
2598   48201            Harris, Texas   4602523.0    0.67668
102     4013        Maricopa, Arizona   4253913.0    0.74400
221     6073    San Diego, California   3302833.0    0.90846
…        …                        …           …        …
2517   48039          Brazoria, Texas    353999.0    0.83867
357    12083          Marion, Florida    348371.0    0.62977
1310   27003        Anoka, Minnesota     347431.0    1.03045
2512   48027              Bell, Texas    342236.0    0.77036
2749   49011              Davis, Utah    340621.0    1.13732

[200 rows x 4 columns]
```

we convert the county number/fips columns to str and merge the two dataframes, then drop the fips column because it's repeated twice ('county' and 'fips')

[19]:
```
counties2h['county'] = counties2h['county'].map(str)
countykir2['fips'] = countykir2['fips'].map(str)
final_df = counties2h.merge(countykir2, how="inner", left_on="county",␣
↪right_on="fips")
final_df = final_df.drop(columns=['fips'])
```

this is how the final dataframe looks like

[20]:
```
final_df
```

[20]:
```
    county              county_name     pop2018  ec_county  income_rank_p25
0     6037  Los Angeles, California  10098052.0    0.73580         0.465747
1    17031           Cook, Illinois   5223719.0    0.75869         0.433307
2    48201            Harris, Texas   4602523.0    0.67668         0.452386
3     4013        Maricopa, Arizona   4253913.0    0.74400         0.430274
4     6073    San Diego, California   3302833.0    0.90846         0.443861
..     …                        …           …        …                …
195  48039          Brazoria, Texas    353999.0    0.83867         0.462893
196  12083          Marion, Florida    348371.0    0.62977         0.397428
197  27003         Anoka, Minnesota    347431.0    1.03045         0.475523
198  48027              Bell, Texas    342236.0    0.77036         0.409022
199  49011              Davis, Utah    340621.0    1.13732         0.452445

[200 rows x 5 columns]
```

We then create a dataframe for the highlighted counties

[21]:
```
highlighted_counties = {'18097': 'Indianapolis', '36061': 'New York',
                        '49035': 'Salt Lake City', '27053': 'Minneapolis',
                        '6037': 'Los Angeles'}
hlcounties_df = final_df.loc[final_df.county.apply(lambda x: x in␣
↪list(highlighted_counties.keys()))]

hlcounties_df['city_name'] = hlcounties_df.county.apply(lambda x:␣
↪highlighted_counties.get(x))
```

```
C:\Users\Kharnifex\AppData\Local\Temp\ipykernel_3832\3526779763.py:6:
SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

[22]:
```
hlcounties_df
```

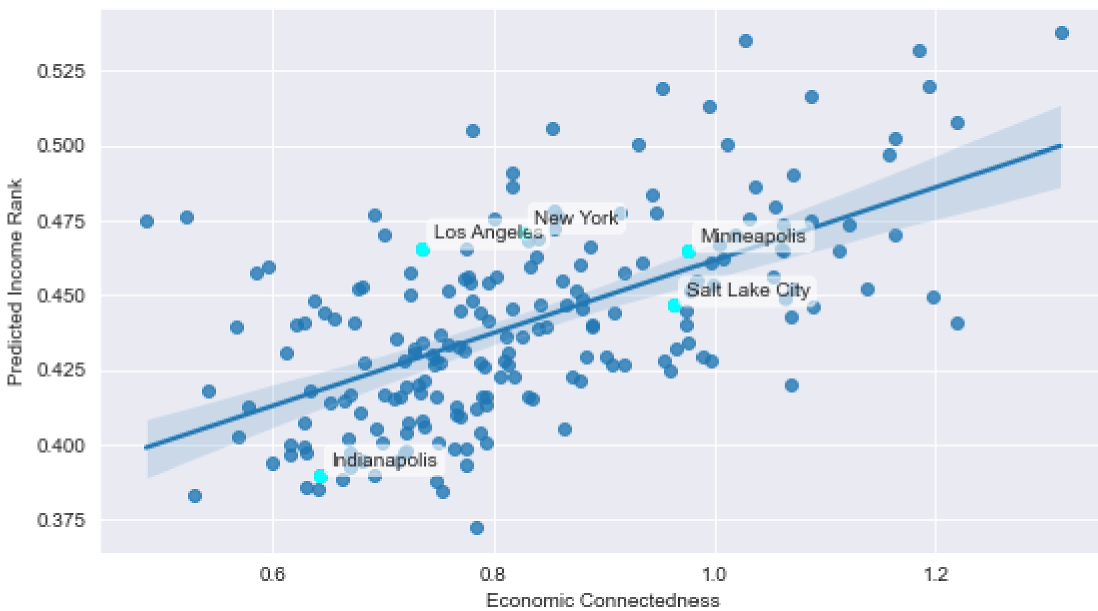
[22]:
```
   county              county_name     pop2018  ec_county  income_rank_p25  \
0    6037  Los Angeles, California  10098052.0    0.73580         0.465747
20  36061       New York, New York   1632480.0    0.82734         0.471015
33  27053      Hennepin, Minnesota   1235478.0    0.97632         0.464788
38  49035          Salt Lake, Utah   1120805.0    0.96395         0.446506
50  18097          Marion, Indiana    944523.0    0.64282         0.389983

         city_name
0      Los Angeles
20        New York
33     Minneapolis
38  Salt Lake City
50    Indianapolis
```

And finally, we visualise the data using a scatterplot

[23]:
```
fig = plt.gcf()
fig.set_size_inches(9, 5)

sns.set_style("darkgrid")
sp2 = sns.regplot(x='ec_county', y='income_rank_p25',
                  data=final_df)

plt.scatter(hlcounties_df.ec_county, hlcounties_df.income_rank_p25,␣
↪color='cyan')
```

```
for row in range(0, hlcounties_df.shape[0]):
    props = dict(boxstyle='round', facecolor='white', alpha=0.6)
    plt.text(x = hlcounties_df.ec_county.iloc[row]+0.01, y = hlcounties_df.
 ↪income_rank_p25.iloc[row]+0.003,
         s = hlcounties_df.city_name.iloc[row], bbox = props)

sp2.set(xlabel = 'Economic Connectedness',
       ylabel = 'Predicted Income Rank')
```

[23]: [Text(0.5, 0, 'Economic Connectedness'), Text(0, 0.5, 'Predicted Income Rank')]



1.5 Q3: Upward Income Mobility, Economic Connectedness, and Median House Income

For this question we're going to need access to data on the median family income in each ZIP code. We have found this data on https://mcdc.missouri.edu/applications/zipcodes/ZIP_codes_2018.xls found in this forum post (https://acsdatacommunity.prb.org/discussion-forum/f/forum/637/zip-code-median-income-data) and converted the file from .xls to .csv (and renamed it to 'zip_codes.csv')

We also use the data from the original dataset for social capital per zip code

[24]:
```
sczipcode_df = pd.read_csv('dataset/social_capital_zip.csv', dtype={'county':
 ↪str})
zipcode_df = pd.read_csv('dataset/zip_codes.csv')
```

[25]:
```
sczipcode_df
```

[25]:

```
         zip  county  num_below_p50  pop2018   ec_zip  ec_se_zip  nbhd_ec_zip  \
0       1001   25013     995.787468    17621  0.88157    0.02422      1.51095
1       1002   25015    1312.117077    30066  1.18348    0.02227      0.97760
2       1003   25015            NaN    11238  1.37536    0.05046          NaN
3       1005   25027     381.519745     4991  1.15543    0.03050      1.46491
4       1007   25015     915.396667    14967  1.19240    0.02046      1.17985
…        …     …              …      …        …        …          …
23023  99901    2130    1192.299809    13818  0.99517    0.01776      0.88014
23024  99921    2198     365.768661     1986  0.87977    0.03071      0.74555
23025  99925    2198     154.513840      927      NaN        NaN          NaN
23026  99926    2198     311.014252     1635  0.87888    0.03618      0.81081
23027  99929    2275     313.282990     2484  1.06344    0.03122      0.88864

       ec_grp_mem_zip  ec_high_zip  ec_high_se_zip  …  \
0             1.10210      1.47136         0.01599  …
1             1.23333      1.62290         0.01500  …
2             1.44359      1.65159         0.02898  …
3             1.30756      1.47733         0.01664  …
4             1.32294      1.56812         0.01364  …
…                   …            …             …  …
23023         0.95456      1.29659         0.01806  …
23024         0.82996      1.18270         0.03593  …
23025             NaN          NaN             NaN  …
23026         0.83409      1.07167         0.04187  …
23027         0.96641      1.32997         0.02900  …

       exposure_grp_mem_high_zip  nbhd_exposure_zip  bias_grp_mem_zip  \
0                        1.45669            1.50590           0.02434
1                        1.53277            1.20282           0.09856
2                        1.57757                NaN           0.02482
3                        1.43769            1.46397           0.00850
4                        1.43019            1.23109          -0.01188
…                              …                  …                 …
23023                    1.09039            0.94762           0.05710
23024                    1.04318            0.81680           0.06010
23025                        NaN                NaN               NaN
23026                    0.92952            0.80694           0.00877
23027                    1.07349            0.88926           0.01350

       bias_grp_mem_high_zip  nbhd_bias_zip  nbhd_bias_high_zip  \
0                   -0.10001       -0.00336            -0.21186
1                   -0.06421        0.18724            -0.24353
2                   -0.05143            NaN                 NaN
3                   -0.07246       -0.00064            -0.11397
4                   -0.11464        0.04162            -0.21283
…                          …              …                   …
23023               -0.14293        0.07122            -0.21950
```

```
23024              -0.08759     0.08723          -0.14339
23025                   NaN         NaN               NaN
23026              -0.07257    -0.00480          -0.09655
23027              -0.14883     0.00069          -0.24887

       clustering_zip  support_ratio_zip  volunteering_rate_zip  \
0            0.105720           0.945260                0.05650
1            0.103400           0.901630                0.14951
2            0.136500           0.769240                0.10501
3            0.105540           0.958370                0.15862
4            0.103910           0.948730                0.13053
…                 …                 …                    …
23023        0.134730           0.997200                0.11883
23024        0.155610           0.997520                0.08404
23025        0.146579           0.992298                0.12396
23026        0.252740           1.000000                0.14291
23027        0.165580           1.000000                0.10700

       civic_organizations_zip
0                     0.010800
1                     0.036880
2                     0.080500
3                     0.021630
4                     0.016900
…                         …
23023                 0.029990
23024                 0.032150
23025                 0.027728
23026                 0.011250
23027                 0.042480

[23028 rows x 23 columns]
```

[26]:
```
zipcode_df
```

[26]:
```
       ZIP Code      Type  State FIPS   Preferred name  \
0           501    unique          36  Holtsville, NY
1           544    unique          36  Holtsville, NY
2           601  standard          72    Adjuntas, PR
3           602  standard          72      Aguada, PR
4           603  standard          72   Aguadilla, PR
…           …         …          …              …
41271     99926    PO box           2  Metlakatla, AK
41272     99927    PO box           2 Point Baker, AK
41273     99928    PO box           2   Ward Cove, AK
41274     99929    PO box           2    Wrangell, AK
41275     99950    PO box           2   Ketchikan, AK
```

```
                                          Alternate names  Population (2018)  \
0                                      IRS Service Center                NaN   
1                                      IRS Service Center                NaN   
2      Colinas Del Gigante, Jard De Adjuntas, Urb San…             17,242   
3      Alts De Aguada, Bo Guaniquilla, Comunidad Las …             38,442   
4      Ramey, Bda Caban, Bda Esteves, Bo Borinquen, B…             48,814   
…                                                   …                  …   
41271                                             NaN              1,635   
41272                                             NaN                 38   
41273                                             NaN                NaN   
41274                                             NaN              2,484   
41275                                 Edna Bay, Kasaan                NaN   

      Housing units (2018)  Median family income (2018)  \
0                      NaN                          NaN   
1                      NaN                          NaN   
2                    7,176                      $14,433   
3                   17,403                      $19,250   
4                   24,311                      $19,718   
…                      …                          …   
41271                  548                      $65,313   
41272                   78                          NaN   
41273                  NaN                          NaN   
41274                1,450                      $71,923   
41275                  NaN                          NaN   

       MFI percentile (2018)   Latitude   Longitude  Land area  Water area  
0                        NaN        NaN         NaN        NaN         NaN  
1                        NaN        NaN         NaN        NaN         NaN  
2                        0.0  18.181000  -66.750000     64.348       0.309  
3                        0.0  18.362000  -67.176003     30.613       1.718  
4                        0.0  18.455000  -67.120003     31.616       0.071  
…                        …        …         …          …         …  
41271                   49.0  55.138000 -131.470001    132.798      82.369  
41272                    NaN  56.238998 -133.457993    227.680       6.950  
41273                    NaN        NaN         NaN        NaN         NaN  
41274                   61.0  56.370998 -131.692993    999.999     246.117  
41275                    NaN        NaN         NaN        NaN         NaN  

[41276 rows x 13 columns]
```

we merge the two dataframes and drop most columns, keeping only the 4 that we need, and then we drop all rows that include NaN values

[27]:
```
zip_df = sczipcode_df.merge(zipcode_df, 'inner', left_on='zip', right_on='ZIP␣
 ↪Code')
```

```
zip_df = zip_df.loc[:,['zip', 'county', 'ec_zip', 'Median family income␣
↪(2018)']]
zip_df = zip_df.dropna(how='any')
zip_df
```

[27]:

```
         zip county   ec_zip Median family income (2018)
0       1001  25013  0.88157                     $88,797
1       1002  25015  1.18348                     $98,977
3       1005  25027  1.15543                    $104,435
4       1007  25015  1.19240                    $108,210
6       1010  25013  0.73856                     $92,841
…        …     …       …                          …
23022  99840   2230  1.11489                     $84,688
23023  99901   2130  0.99517                     $85,295
23024  99921   2198  0.87977                     $78,958
23026  99926   2198  0.87888                     $65,313
23027  99929   2275  1.06344                     $71,923

[18895 rows x 4 columns]
```

We then merge the dataframe from the previous question with the one we just created so we can have the projected income rank for the 25th percentile per county, and multiply all values of that column by 100 so we can more easily visualise it later

[28]:

```
thirdq_df = zip_df.merge(countykir2, 'inner', left_on='county', right_on='fips')
thirdq_df = thirdq_df.drop(columns=['fips'])
thirdq_df.income_rank_p25 = thirdq_df.income_rank_p25.apply(lambda x: x*100)
thirdq_df
```

[28]:

```
         zip county   ec_zip Median family income (2018)  income_rank_p25
0       1001  25013  0.88157                     $88,797        44.087276
1       1010  25013  0.73856                     $92,841        44.087276
2       1013  25013  0.69744                     $50,963        44.087276
3       1020  25013  0.72701                     $70,974        44.087276
4       1022  25013  0.79394                     $51,650        44.087276
…        …     …       …                          …                  …
18890  99840   2230  1.11489                     $84,688        54.440475
18891  99901   2130  0.99517                     $85,295        45.691451
18892  99921   2198  0.87977                     $78,958        37.976360
18893  99926   2198  0.87888                     $65,313        37.976360
18894  99929   2275  1.06344                     $71,923        43.971640

[18895 rows x 5 columns]
```

We create bins in order to better visualise our data

[29]:

```
beans = pd.cut(thirdq_df.income_rank_p25, bins=np.array([0,38,42,44,48,␣
↪float('inf')]),
```

```
                 labels=['<38', '38-41', '41-44', '44-48', '>48'])
thirdq_df['bin'] = beans
thirdq_df
```

```
[29]:          zip  county   ec_zip Median family income (2018)  income_rank_p25  \
0       1001   25013  0.88157                     $88,797        44.087276
1       1010   25013  0.73856                     $92,841        44.087276
2       1013   25013  0.69744                     $50,963        44.087276
3       1020   25013  0.72701                     $70,974        44.087276
4       1022   25013  0.79394                     $51,650        44.087276
…       …      …       …                          …                 …
18890  99840    2230  1.11489                     $84,688        54.440475
18891  99901    2130  0.99517                     $85,295        45.691451
18892  99921    2198  0.87977                     $78,958        37.976360
18893  99926    2198  0.87888                     $65,313        37.976360
18894  99929    2275  1.06344                     $71,923        43.971640

         bin
0      44-48
1      44-48
2      44-48
3      44-48
4      44-48
…      …
18890    >48
18891  44-48
18892    <38
18893    <38
18894  41-44

[18895 rows x 6 columns]
```

We notice that the income column has string values with $ and commas, so we remove those characters from all strings and convert them to integers

```
[30]: thirdq_df['Median family income (2018)'] = thirdq_df['Median family income␣
 ↪(2018)']\
                                             .apply(lambda x: x.replace('$',␣
 ↪''))\
                                             .apply(lambda x: x.replace(',',␣
 ↪'')).astype(np.int64)
```

We then keep only rows where the median family income is between 30000 and 100000

```
[31]: thirdq_df = thirdq_df.loc[(thirdq_df['Median family income (2018)'] >= 30000) &␣
 ↪(thirdq_df['Median family income (2018)'] <= 100000)]
thirdq_df
```

[31]:
```
         zip county   ec_zip  Median family income (2018)  income_rank_p25  \
0       1001  25013  0.88157                        88797        44.087276
1       1010  25013  0.73856                        92841        44.087276
2       1013  25013  0.69744                        50963        44.087276
3       1020  25013  0.72701                        70974        44.087276
4       1022  25013  0.79394                        51650        44.087276
…        …      …        …                          …               …
18890  99840   2230  1.11489                        84688        54.440475
18891  99901   2130  0.99517                        85295        45.691451
18892  99921   2198  0.87977                        78958        37.976360
18893  99926   2198  0.87888                        65313        37.976360
18894  99929   2275  1.06344                        71923        43.971640

         bin
0      44-48
1      44-48
2      44-48
3      44-48
4      44-48
…        …
18890    >48
18891  44-48
18892    <38
18893    <38
18894  41-44

[15687 rows x 6 columns]
```

our data is ready to be visualised in a scatterplot

[32]:
```python
fig = plt.gcf()
fig.set_size_inches(13, 7)

sp3 = sns.scatterplot(x='Median family income (2018)', y='ec_zip', s=50,␣
 ↪alpha=0.8,
                hue=beans, data=thirdq_df)

sp3.set(xlabel = 'Median Family Income',
       ylabel = 'Economic Connectedness',
       title = 'Association between upward income mobility and economic␣
 ↪connectedness')

legend = plt.legend(title='Upward Mobility', loc=4)

plt.setp(legend.get_title(), fontsize = 'x-large')
```

[32]:
```
[None, None]
```

1.6 Q4: Friending Bias and Exposure by High School

First of all, we import the high school data from the social capital dataset into a dataframe

[33]:
```
highschool_df = pd.read_csv('dataset/social_capital_high_school.csv')
highschool_df
```

[33]:
```
      high_school                      high_school_name    zip  county  \
0        00000044           Holy Spirit Catholic School  35405    1125
1        00000226              John Carroll Catholic HS  35209    1073
2        00000237       Holy Family Cristo Rey Catholic HS  35218    1073
3        00000714  Montgomery Catholic Preparatory School  36116    1101
4        00000758              St Paul's Episcopal School  36608    1097
…             …                                     …      …       …
17520    Y2121679     St Agnes Academy-St Dominic School  38117   47157
17521    Z0516931                          Sayre School  40507   21067
17522    Z1326859           Fort Worth Christian School  76180   48439
17523    Z1326892                Second Baptist School  77057   48201
17524    Z1328448                    The Kinkaid School  77024   48201

       students_9_to_12  ec_own_ses_hs  ec_own_ses_se_hs  ec_parent_ses_hs  \
0                   158            NaN               NaN               NaN
1                   538        1.52901           0.04220           1.43847
2                   229        0.66359           0.07105               NaN
3                   363        1.56551           0.05799               NaN
4                   409        1.62628           0.04533           1.57592
…                     …              …                 …                 …
```

```
17520             350          NaN           NaN           NaN
17521             258          NaN           NaN           NaN
17522             327          NaN           NaN           NaN
17523             338          NaN           NaN           NaN
17524             588          NaN           NaN           NaN

       ec_parent_ses_se_hs  ec_high_own_ses_hs  …  ec_high_parent_ses_hs  \
0                      NaN                 NaN  …                    NaN
1                  0.05073             1.64439  …                1.46086
2                      NaN             0.87627  …                    NaN
3                      NaN             1.60898  …                    NaN
4                  0.05254             1.72722  …                1.60072
…                        …                 …  …                      …
17520                  NaN                 NaN  …                    NaN
17521                  NaN                 NaN  …                    NaN
17522                  NaN                 NaN  …                    NaN
17523                  NaN                 NaN  …                    NaN
17524                  NaN                 NaN  …                    NaN

       ec_high_parent_ses_se_hs  exposure_own_ses_hs  exposure_parent_ses_hs  \
0                           NaN                  NaN                     NaN
1                       0.04742              1.50707                 1.44259
2                           NaN              0.65517                     NaN
3                           NaN              1.49000                     NaN
4                       0.04730              1.62275                 1.57514
…                             …                    …                       …
17520                       NaN                  NaN                     NaN
17521                       NaN                  NaN                     NaN
17522                       NaN                  NaN                     NaN
17523                       NaN                  NaN                     NaN
17524                       NaN                  NaN                     NaN

       bias_own_ses_hs  bias_parent_ses_hs  bias_high_own_ses_hs  \
0                  NaN                 NaN                   NaN
1             -0.01456             0.00285              -0.09112
2             -0.01286                 NaN              -0.33747
3             -0.05068                 NaN              -0.07985
4             -0.00217            -0.00050              -0.06438
…                    …                   …                     …
17520              NaN                 NaN                   NaN
17521              NaN                 NaN                   NaN
17522              NaN                 NaN                   NaN
17523              NaN                 NaN                   NaN
17524              NaN                 NaN                   NaN

       bias_high_parent_ses_hs  clustering_hs  volunteering_rate_hs
0                          NaN       0.693142              0.086807
```

```
1                       -0.01266      0.604580            0.069540
2                            NaN      0.686860            0.051010
3                            NaN      0.673730            0.042280
4                       -0.01624      0.623290            0.060610
…                              …             …                   …
17520                        NaN      0.644070            0.077204
17521                        NaN      0.740327            0.092056
17522                        NaN      0.680769            0.053181
17523                        NaN      0.692155            0.050045
17524                        NaN      0.643250            0.047230

[17525 rows x 21 columns]
```

We convert the two columns that we'll visualise to percentages and change the highschool code to a string value

[34]:
```
highschool_df['bias_parent_ses_hs'] = highschool_df['bias_parent_ses_hs'].
 ↪apply(lambda x: x*100)
highschool_df['exposure_parent_ses_hs'] =␣
 ↪highschool_df['exposure_parent_ses_hs'].apply(lambda x: x*50)
highschool_df['high_school'] = highschool_df['high_school'].map(str)
```

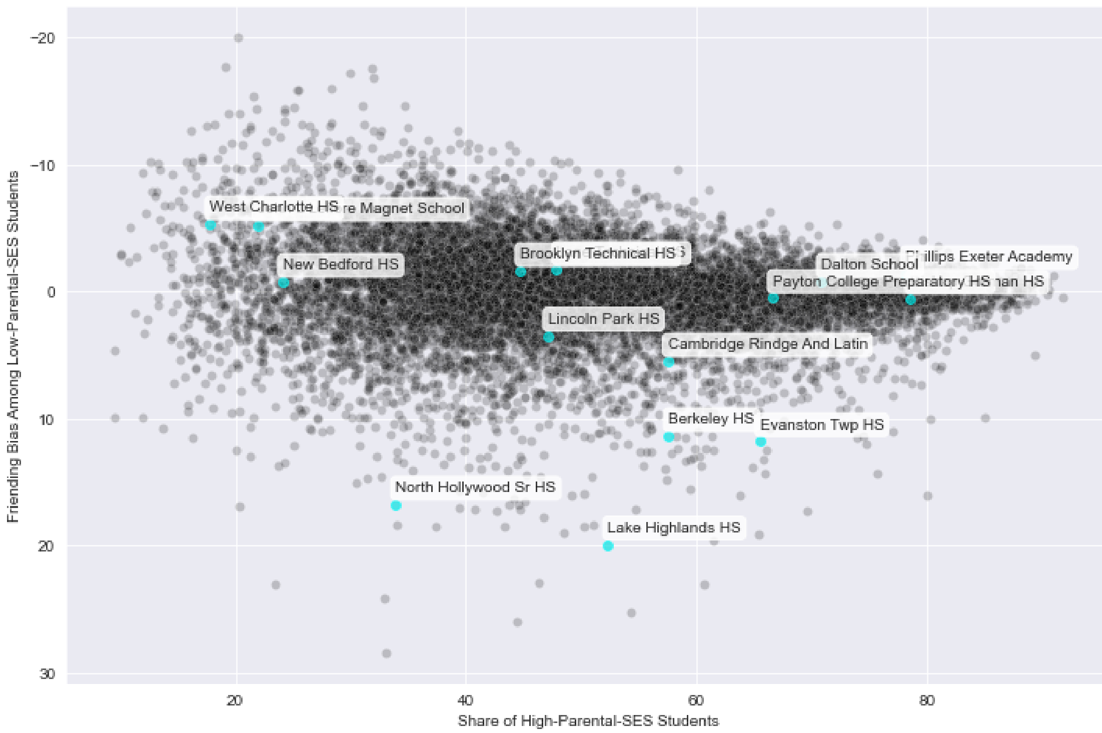
We create a dataframe with the schools we want to highlight

[35]:
```
highlighted_schools = ['00941729', '060474000432', '170993000942',␣
 ↪'170993001185', '170993003989',
                       '171449001804', '250327000436', '360009101928',␣
 ↪'370297001285', '483702004138',
                       '250843001336', '062271003230',␣
 ↪'010237000962','00846981', '00852124']
highlighted_df = highschool_df.loc[highschool_df.high_school.apply(lambda x: x␣
 ↪in highlighted_schools)]
```

We then visualise the data using another scatterplot

[36]:
```
fig = plt.gcf()
fig.set_size_inches(12, 8)

sp4 = sns.scatterplot( x='exposure_parent_ses_hs', y='bias_parent_ses_hs',
                     alpha=0.2, color='black', data = highschool_df)

plt.scatter(highlighted_df.exposure_parent_ses_hs ,highlighted_df.
 ↪bias_parent_ses_hs , color='cyan', alpha=0.6)

for row in range(0, highlighted_df.shape[0]):
    props = dict(boxstyle='round', facecolor='white', alpha=0.8)
    plt.text(x = highlighted_df.exposure_parent_ses_hs.iloc[row], y =␣
 ↪highlighted_df.bias_parent_ses_hs.iloc[row]-1,
```

```
        s = highlighted_df.high_school_name.iloc[row], bbox=props)

sp4.set(xlabel = 'Share of High-Parental-SES Students',
        ylabel = 'Friending Bias Among Low-Parental-SES Students')

sp4.invert_yaxis()
```



1.7 Q5: Friending Bias vs. Racial Diversity

For this question we use the 2-year College Representativeness data from https://datacatalog.urban.org/dataset/racial-and-ethnic-representativeness-us-postsecondary-education-institutions (renamed the csv file for clarity) for the collegedata dataframe as well as the original social capital data for colleges for college_df

As for the neighborhoods we use the zip-code dataset from the social capital one and the DP05 ACS DEMOGRAPHIC AND HOUSING ESTIMATES dataset found on https://data.census.gov/cedsci/table?g=0100000US%248600000&tid=ACSDP5Y2020.DP05 with the Select 860 - 5-Digit ZCTA filter on the geos tab (note: that tab can be found only on 5-year estimates so we have to download that and then just pick the 2018 csv file) renamed from 'ACSDP5Y2018.DP05-Data.csv' to 'neighborhood_data.csv'

[37]:
```
college_df = pd.read_csv('dataset/social_capital_college.csv')
neighborhood_df = pd.read_csv('dataset/social_capital_zip.csv')
collegedata = pd.read_csv('dataset/college_data.csv')
```

```
neighborhooddata = pd.read_csv('dataset/neighborhood_data.csv', header=1)
collegedata = collegedata[collegedata['year'] == 2013]
```

```
c:\users\kharnifex\appdata\local\programs\python\python38\lib\site-
packages\IPython\core\interactiveshell.py:3012: DtypeWarning:

Columns (4,5,8,9,12,13,19,20,31,32,35,36,39,40,43,44,47,48,60,61,75,76,83,84,95,
96,100,101,103,104,107,108,115,116,119,120,123,124,131,132,251,252,279,280,284,2
85,304,305,362,363,364,365,366,367,368,369,374,375,376,377,378,379,380,381,382,3
83,384,385,386,387,388,389,390,391,392,393,394,395,396,397,398,399,400,401,402,4
03,404,405,406,407,408,409,410,411,412,413,414,415,416,417,418,419,420,421,422,4
23,424,425,430,431,432,433,434,435,436,437,438,439,440,441,442,443,444,445,446,4
47,448,449,450,451,452,453,458,459,460,461,462,463,464,465,490,491,492,493,494,4
95,496,497,498,499,500,501,502,503,504,505,506,507,508,509,510,511,512,513,514,5
15,516,517,518,519,520,521,522,523,524,525,526,527,528,529,530,531,532,533,534,5
35,536,537,538,539,540,541,542,543,544,545,546,547,548,549,550,551,552,553,554,5
55,556,557,558,559,560,561,562,563,564,565,566,567,568,569,570,571,572,573,574,5
75,576,577,578,579,580,581,582,583,584,585,586,587,588,589,590,591,592,593,594,5
95,596,597,598,599,600,601,602,603,604,605,610,611,612,613,614,615,616,617,618,6
19,620,621,622,623,624,625,626,627,628,629,630,631,632,633,638,639,640,641,642,6
43,644,645,646,647,648,649,650,651,652,653,654,655,656,657,658,659,660,661,662,6
63,664,665,666,667,668,669,670,671,672,673,674,675,676,677,678,679,680,681,682,6
83,684,685,686,687,688,689,690,691,692,693,694,695,696,697,706,707,708,709,710,7
11,712,713) have mixed types.Specify dtype option on import or set
low_memory=False.
```

[38]:
```
college_df
```

[38]:
```
      college                         college_name    zip  county  \
0      100200                Alabama A & M University  35762    1089
1      100300                     Faulkner University  36109    1101
2      100400                University of Montevallo  35115    1117
3      100500                Alabama State University  36104    1101
4      100700        Central Alabama Community College  35010    1123
…        …                                        …     …      …
2581  4254400  Arkansas State University-Mountain Home  72653    5005
2582  4263400             Florida Polytechnic University  33805   12105
2583  4263600               Northeast Lakeview College  78145   48029
2584  4281700                         Compton College  90221    6037
2585  4283700            Oregon Coast Community College  97366   41041

      mean_students_per_cohort  ec_own_ses_college  ec_own_ses_se_college  \
0                   943.666667             0.85678                0.02233
1                   227.666667             1.30964                0.04869
2                   494.000000             1.42378                0.03040
3                          NaN             0.77916                0.01937
4                          NaN             0.72742                0.03504
```

```
…                         …                 …                     …
2581                      NaN         0.88695               0.04674
2582                      NaN             NaN                   NaN
2583                      NaN         1.28254               0.05277
2584                      NaN         0.71178               0.06780
2585                      NaN         0.69457               0.07714

      ec_parent_ses_college  ec_parent_ses_se_college  \
0                   0.67629                   0.03241
1                   1.26671                   0.05812
2                   1.15413                   0.03638
3                   0.67090                   0.03038
4                   0.77238                   0.04497
…                       …                       …
2581                0.52927                   0.05098
2582                1.20327                   0.09919
2583                1.17784                   0.06483
2584                    NaN                       NaN
2585                    NaN                       NaN

      ec_high_own_ses_college  …  ec_high_parent_ses_se_college  \
0                     1.12202  …                        0.03498
1                     1.54639  …                        0.05134
2                     1.57365  …                        0.03395
3                     1.04811  …                        0.03201
4                     0.98888  …                        0.04984
…                         …   …                        …
2581                  1.00103  …                        0.05764
2582                      NaN  …                        0.09509
2583                  1.41132  …                        0.06000
2584                  0.81637  …                            NaN
2585                  0.80913  …                            NaN

      exposure_own_ses_college  exposure_parent_ses_college  \
0                      0.84662                      0.65090
1                      1.23776                      1.20183
2                      1.41664                      1.17101
3                      0.75162                      0.65297
4                      0.76579                      0.76786
…                        …                           …
2581                   0.89316                      0.49553
2582                       NaN                      1.19730
2583                   1.36033                      1.17411
2584                   0.72474                          NaN
2585                   0.71267                          NaN

      bias_own_ses_college  bias_parent_ses_college  \
```

```
0                  -0.01200                -0.03900
1                  -0.05807                -0.05398
2                  -0.00504                 0.01442
3                  -0.03664                -0.02747
4                   0.05010                -0.00589
…                     …                     …
2581                0.00695                -0.06810
2582                    NaN                -0.00499
2583                0.05718                -0.00318
2584                0.01789                     NaN
2585                0.02539                     NaN

      bias_high_own_ses_college  bias_high_parent_ses_college  \
0                      -0.32529                      -0.14036
1                      -0.24935                      -0.12001
2                      -0.11083                      -0.05979
3                      -0.39448                      -0.12802
4                      -0.29133                      -0.13139
…                         …                           …
2581                   -0.12077                      -0.15805
2582                        NaN                      -0.03957
2583                   -0.03748                      -0.06948
2584                   -0.12643                           NaN
2585                   -0.13536                           NaN

      clustering_college  support_ratio_college  volunteering_rate_college
0                0.24470                0.99483                    0.03256
1                0.40754                0.99481                    0.03336
2                0.30921                0.99683                    0.09566
3                0.23222                0.99485                    0.02150
4                0.34104                0.99271                    0.02922
…                  …                      …                          …
2581             0.32144                0.99446                    0.06755
2582             0.48909                0.99920                    0.04523
2583             0.24113                0.90760                    0.03251
2584             0.21260                0.82709                    0.02312
2585             0.34485                0.97277                    0.12013

[2586 rows x 22 columns]
```

[39]:
```
collegedata
```

[39]:
```
       unitid  year fips_ipeds                                inst_name  \
4      100760  2013    Alabama         Central Alabama Community College
13     101028  2013    Alabama     Chattahoochee Valley Community College
26     101143  2013    Alabama          Enterprise State Community College
35     101161  2013    Alabama  James H Faulkner State Community College
```

44	101240	2013	Alabama	Gadsden State Community College
…	…	…	…	…
16306	240620	2013	Wyoming	Laramie County Community College
16315	240657	2013	Wyoming	Northwest College
16324	240666	2013	Wyoming	Sheridan College
16333	240693	2013	Wyoming	Western Wyoming Community College
16342	240718	2013	Wyoming	Wyotech-Laramie

	slevel	twocat	public	forprofit	total_enrollment \
4	2-year	Public 2-year	1	0	1779.0
13	2-year	Public 2-year	1	0	1769.0
26	2-year	Public 2-year	1	0	1984.0
35	2-year	Public 2-year	1	0	4182.0
44	2-year	Public 2-year	1	0	5480.0
…	…	…	…	…	…
16306	2-year	Public 2-year	1	0	3516.0
16315	2-year	Public 2-year	1	0	1624.0
16324	2-year	Public 2-year	1	0	2108.0
16333	2-year	Public 2-year	1	0	2792.0
16342	2-year	For-Profit 2-year	0	1	1462.0

	col_white	…	dif_asian	col_amind	mkt_amind	dif_amind	col_pacis \
4	67.84710	…	0.008670	0.224845	0.278752	-0.053907	0.112423
13	46.74958	…	-0.744371	0.339175	0.181952	0.157222	0.282646
26	64.11290	…	0.485406	0.554435	0.462555	0.091881	0.151210
35	71.04256	…	-0.461236	1.865136	0.956258	0.908878	0.119560
44	67.91971	…	-0.177262	0.748175	0.372870	0.375305	0.145985
…	…	…	…	…	…	…	…
16306	80.03413	…	-0.190118	1.023891	0.999230	0.024661	0.284414
16315	83.49754	…	-0.327457	0.923645	0.826607	0.097038	0.061576
16324	88.23530	…	0.064210	1.755218	1.396530	0.358688	0.047438
16333	82.80802	…	-0.171320	0.644699	0.327130	0.317570	0.143266
16342	69.69904	…	-3.295658	3.898769	0.535388	3.363381	0.547196

	mkt_pacis	dif_pacis	col_twora	mkt_twora	dif_twora
4	0.000000	0.112423	0.056211	0.754717	-0.698506
13	0.055543	0.227102	1.074053	1.200885	-0.126832
26	0.018287	0.132923	1.764113	1.748026	0.016087
35	0.016984	0.102576	1.626016	1.074118	0.551898
44	0.101344	0.044641	1.897810	1.437941	0.459869
…	…	…	…	…	…
16306	0.044590	0.239824	0.341297	2.067372	-1.726075
16315	0.000000	0.061576	2.463054	1.517067	0.945987
16324	0.126957	-0.079519	1.992410	1.855994	0.136416
16333	0.030945	0.112322	1.683381	5.127978	-3.444597
16342	0.073008	0.474188	2.667579	2.202393	0.465186

```
[1972 rows x 30 columns]
```

We then calculate the HHI for each row of the collegedata dataframe

[40]:
```
collegedata['HHI'] = collegedata.apply(lambda x: 1 - (np.square(x.col_white/
↪100) + np.square(x.col_amind/100) +
                                        np.square(x.col_asian/100) + np.square(x.
↪col_hispa/100) +
                                        np.square(x.col_black/100) + np.square(x.
↪col_pacis/100) + np.square(x.col_twora/100)), axis=1)
```

we narrow down the columns we want to keep for the two dataframes and merge them into one, then we multiply the friending bias by 100 so that we have a percentage instead of a decimal

[41]:
```
collegedata = collegedata.loc[:,['inst_name', 'HHI']]
college_df = college_df.loc[:,['college', 'college_name', 'zip', 'county',␣
↪'bias_own_ses_college', 'mean_students_per_cohort']]
col_df = collegedata.merge(college_df, 'inner', left_on= 'inst_name',␣
↪right_on='college_name')
col_df = col_df.drop(columns=['college_name'])
col_df['bias_own_ses_college'] = col_df['bias_own_ses_college'].apply(lambda x:␣
↪x*100)
col_df.drop_duplicates(subset='inst_name', inplace=True)
col_df.dropna(inplace=True)
```

we arrange the dataframe by dividing it into ventiles based on the HHI column

[42]:
```
func = np.vectorize(lambda x: str(x) + '%')
bins = pd.cut(col_df.HHI, bins = 20, labels = func(np.arange(0,100,5)))
col_df['category'] = bins
```

After that, we calculate the first form of weighted HHI. (we will have to divide the sum of weighted HHIs by the sum of mean students per cohort in each bin, but we will be doing that later)

We then want to basically run the following SQL Query on our Dataframe:

‘SELECT category, SUM(weighted_HHI), SUM(mean_students_per_cohort), AVG(bias_own_ses_college) FROM col_df GROUP BY category’

the mess below is the only way i actually managed to make that work

[43]:
```
col_df['weighted_hhi'] = col_df.apply((lambda x: x.HHI * x.
↪mean_students_per_cohort), axis=1)

df_1 = col_df.groupby('category')\
                .sum(numeric_only=True).reset_index()\
                .loc[:,['category', 'mean_students_per_cohort']]

df_2 = col_df.groupby('category')\
            .mean(numeric_only=True).reset_index()\
```

```
            .loc[:,['category', 'bias_own_ses_college']]

df_3 = col_df.groupby('category')\
                .sum(numeric_only=True).reset_index()\
                .loc[:,['category', 'weighted_hhi']]

df_4 = df_2.merge(df_3, 'inner', left_on = 'category', right_on='category')

final_col_df = df_4.merge(df_1, 'inner', left_on = 'category',␣
 ↪right_on='category')
```

After all that, we calculate the actual weighted hhi for each row (bin), which as stated before is the sum of original weighted HHIs divided by the sum of mean_students_per_cohort

[44]:
```
final_col_df['weighted_hhi']= final_col_df.weighted_hhi / final_col_df.
 ↪mean_students_per_cohort
```

This is how the college dataframe looks like in the end

[45]:
```
final_col_df
```

[45]:

	category	bias_own_ses_college	weighted_hhi	mean_students_per_cohort
0	0%	-2.963625	0.087307	3022.166667
1	5%	-2.779067	0.137573	39334.833333
2	10%	-2.012700	0.178627	5684.000000
3	15%	-0.145636	0.203113	37676.000000
4	20%	-0.734423	0.261555	70857.666667
5	25%	-0.388647	0.293442	101430.000000
6	30%	1.684610	0.337133	195517.000000
7	35%	1.558242	0.376778	100919.000000
8	40%	1.627625	0.418815	67892.333333
9	45%	4.381886	0.453770	70503.500000
10	50%	3.041218	0.496116	84063.833333
11	55%	4.610663	0.532109	88848.166667
12	60%	5.449333	0.570980	211195.166667
13	65%	5.535104	0.608435	229236.166667
14	70%	6.269547	0.654398	146802.000000
15	75%	6.437273	0.687581	148439.833333
16	80%	6.006326	0.730718	213973.166667
17	85%	4.505556	0.772389	94127.333333
18	90%	0.148333	0.810641	2610.166667
19	95%	3.493000	0.844320	1769.666667

After that, we move on to the datasets for zip-codes.

[46]:
```
neighborhood_df
```

[46]:

	zip	county	num_below_p50	pop2018	ec_zip	ec_se_zip	\
0	1001	25013.0	995.787468	17621	0.88157	0.02422	

```
1        1002  25015.0    1312.117077    30066  1.18348    0.02227
2        1003  25015.0            NaN    11238  1.37536    0.05046
3        1005  25027.0     381.519745     4991  1.15543    0.03050
4        1007  25015.0     915.396667    14967  1.19240    0.02046
…       …      …             …       …      …       …
23023  99901   2130.0    1192.299809    13818  0.99517    0.01776
23024  99921   2198.0     365.768661     1986  0.87977    0.03071
23025  99925   2198.0     154.513840      927      NaN        NaN
23026  99926   2198.0     311.014252     1635  0.87888    0.03618
23027  99929   2275.0     313.282990     2484  1.06344    0.03122

       nbhd_ec_zip  ec_grp_mem_zip  ec_high_zip  ec_high_se_zip  …  \
0          1.51095         1.10210      1.47136         0.01599  …
1          0.97760         1.23333      1.62290         0.01500  …
2              NaN         1.44359      1.65159         0.02898  …
3          1.46491         1.30756      1.47733         0.01664  …
4          1.17985         1.32294      1.56812         0.01364  …
…             …              …             …               …   …
23023      0.88014         0.95456      1.29659         0.01806  …
23024      0.74555         0.82996      1.18270         0.03593  …
23025          NaN             NaN          NaN             NaN  …
23026      0.81081         0.83409      1.07167         0.04187  …
23027      0.88864         0.96641      1.32997         0.02900  …

       exposure_grp_mem_high_zip  nbhd_exposure_zip  bias_grp_mem_zip  \
0                        1.45669            1.50590           0.02434
1                        1.53277            1.20282           0.09856
2                        1.57757                NaN           0.02482
3                        1.43769            1.46397           0.00850
4                        1.43019            1.23109          -0.01188
…                             …                  …               …
23023                    1.09039            0.94762           0.05710
23024                    1.04318            0.81680           0.06010
23025                        NaN                NaN               NaN
23026                    0.92952            0.80694           0.00877
23027                    1.07349            0.88926           0.01350

       bias_grp_mem_high_zip  nbhd_bias_zip  nbhd_bias_high_zip  \
0                   -0.10001       -0.00336            -0.21186
1                   -0.06421        0.18724            -0.24353
2                   -0.05143            NaN                 NaN
3                   -0.07246       -0.00064            -0.11397
4                   -0.11464        0.04162            -0.21283
…                         …            …                   …
23023               -0.14293        0.07122            -0.21950
23024               -0.08759        0.08723            -0.14339
23025                    NaN            NaN                 NaN
```

```
23026             -0.07257       -0.00480            -0.09655
23027             -0.14883        0.00069            -0.24887

       clustering_zip  support_ratio_zip  volunteering_rate_zip  \
0            0.105720           0.945260                0.05650
1            0.103400           0.901630                0.14951
2            0.136500           0.769240                0.10501
3            0.105540           0.958370                0.15862
4            0.103910           0.948730                0.13053
…                 …                 …                    …
23023        0.134730           0.997200                0.11883
23024        0.155610           0.997520                0.08404
23025        0.146579           0.992298                0.12396
23026        0.252740           1.000000                0.14291
23027        0.165580           1.000000                0.10700

       civic_organizations_zip
0                     0.010800
1                     0.036880
2                     0.080500
3                     0.021630
4                     0.016900
…                          …
23023                 0.029990
23024                 0.032150
23025                 0.027728
23026                 0.011250
23027                 0.042480

[23028 rows x 23 columns]
```

[47]:
```
neighborhooddata
```

[47]:
```
            Geography Geographic Area Name  \
0      8600000US00601          ZCTA5 00601
1      8600000US00602          ZCTA5 00602
2      8600000US00603          ZCTA5 00603
3      8600000US00606          ZCTA5 00606
4      8600000US00610          ZCTA5 00610
…                 …                    …
33115  8600000US99923          ZCTA5 99923
33116  8600000US99925          ZCTA5 99925
33117  8600000US99926          ZCTA5 99926
33118  8600000US99927          ZCTA5 99927
33119  8600000US99929          ZCTA5 99929

       Estimate!!SEX AND AGE!!Total population  \
```

```
0                                        17242
1                                        38442
2                                        48814
3                                         6437
4                                        27073
…                                        …
33115                                       15
33116                                      927
33117                                     1635
33118                                       38
33119                                     2484

       Annotation of Estimate!!SEX AND AGE!!Total population  \
0                                                  NaN
1                                                  NaN
2                                                  NaN
3                                                  NaN
4                                                  NaN
…                                                …
33115                                              NaN
33116                                              NaN
33117                                              NaN
33118                                              NaN
33119                                              NaN

      Margin of Error!!SEX AND AGE!!Total population  \
0                                              314
1                                              150
2                                              749
3                                              304
4                                              205
…                                            …
33115                                           22
33116                                           98
33117                                          122
33118                                           30
33119                                        *****

      Annotation of Margin of Error!!SEX AND AGE!!Total population  \
0                                                  NaN
1                                                  NaN
2                                                  NaN
3                                                  NaN
4                                                  NaN
…                                                …
33115                                              NaN
33116                                              NaN
```

```
33117                                                 NaN
33118                                                 NaN
33119                                               *****

       Estimate!!SEX AND AGE!!Total population!!Male  \
0                                                   8426
1                                                  18842
2                                                  23939
3                                                   3212
4                                                  13112
…                                                  …
33115                                                  0
33116                                                526
33117                                                882
33118                                                 20
33119                                               1302

       Annotation of Estimate!!SEX AND AGE!!Total population!!Male  \
0                                                   NaN
1                                                   NaN
2                                                   NaN
3                                                   NaN
4                                                   NaN
…                                                  …
33115                                               NaN
33116                                               NaN
33117                                               NaN
33118                                               NaN
33119                                               NaN

      Margin of Error!!SEX AND AGE!!Total population!!Male  \
0                                                   159
1                                                    60
2                                                   366
3                                                   187
4                                                    82
…                                                  …
33115                                                 9
33116                                                64
33117                                                68
33118                                                18
33119                                                76

      Annotation of Margin of Error!!SEX AND AGE!!Total population!!Male  …  \
0                                                   NaN                   …
1                                                   NaN                   …
2                                                   NaN                   …
```

```
3                                                         NaN                 …
4                                                         NaN                 …
…                                                       …                   …
33115                                                     NaN                 …
33116                                                     NaN                 …
33117                                                     NaN                 …
33118                                                     NaN                 …
33119                                                     NaN                 …

       Percent Annotation of Estimate!!CITIZEN, VOTING AGE POPULATION!!Citizen,
18 and over population  \
0                                                         NaN
1                                                         NaN
2                                                         NaN
3                                                         NaN
4                                                         NaN
…                                                       …
33115                                                     NaN
33116                                                     NaN
33117                                                     NaN
33118                                                     NaN
33119                                                     NaN

       Percent Estimate!!CITIZEN, VOTING AGE POPULATION!!Citizen, 18 and over
population!!Male  \
0                                                        48.1
1                                                        48.7
2                                                        48.1
3                                                        49.3
4                                                        47.6
…                                                       …
33115                                                     0.0
33116                                                    55.5
33117                                                    53.6
33118                                                    52.6
33119                                                    51.3

      Percent Margin of Error!!CITIZEN, VOTING AGE POPULATION!!Citizen, 18 and
over population!!Male  \
0                                                         0.6
1                                                         0.2
2                                                         0.6
3                                                         1.5
4                                                         0.2
…                                                       …
33115                                                    60.1
33116                                                     3.7
```

```
33117                                                         2.9
33118                                                        26.3
33119                                                         2.9

      Percent Annotation of Margin of Error!!CITIZEN, VOTING AGE
POPULATION!!Citizen, 18 and over population!!Male  \
0                                                             NaN
1                                                             NaN
2                                                             NaN
3                                                             NaN
4                                                             NaN
…                                                         …
33115                                                         NaN
33116                                                         NaN
33117                                                         NaN
33118                                                         NaN
33119                                                         NaN

      Percent Annotation of Estimate!!CITIZEN, VOTING AGE POPULATION!!Citizen,
18 and over population!!Male  \
0                                                             NaN
1                                                             NaN
2                                                             NaN
3                                                             NaN
4                                                             NaN
…                                                         …
33115                                                         NaN
33116                                                         NaN
33117                                                         NaN
33118                                                         NaN
33119                                                         NaN

      Percent Estimate!!CITIZEN, VOTING AGE POPULATION!!Citizen, 18 and over
population!!Female  \
0                                                            51.9
1                                                            51.3
2                                                            51.9
3                                                            50.7
4                                                            52.4
…                                                         …
33115                                                       100.0
33116                                                        44.5
33117                                                        46.4
33118                                                        47.4
33119                                                        48.7

      Percent Annotation of Estimate!!CITIZEN, VOTING AGE POPULATION!!Citizen,
```

```
18 and over population!!Female  \
0                                                     NaN
1                                                     NaN
2                                                     NaN
3                                                     NaN
4                                                     NaN
…                                                   …
33115                                                 NaN
33116                                                 NaN
33117                                                 NaN
33118                                                 NaN
33119                                                 NaN

      Percent Margin of Error!!CITIZEN, VOTING AGE POPULATION!!Citizen, 18 and
over population!!Female  \
0                                                     0.6
1                                                     0.2
2                                                     0.6
3                                                     1.5
4                                                     0.2
…                                                   …
33115                                                60.1
33116                                                 3.7
33117                                                 2.9
33118                                                26.3
33119                                                 2.9

       Percent Annotation of Margin of Error!!CITIZEN, VOTING AGE
POPULATION!!Citizen, 18 and over population!!Female  \
0                                                     NaN
1                                                     NaN
2                                                     NaN
3                                                     NaN
4                                                     NaN
…                                                   …
33115                                                 NaN
33116                                                 NaN
33117                                                 NaN
33118                                                 NaN
33119                                                 NaN

      Unnamed: 714
0              NaN
1              NaN
2              NaN
3              NaN
4              NaN
```

```
…            …
33115          NaN
33116          NaN
33117          NaN
33118          NaN
33119          NaN

[33120 rows x 715 columns]
```

These are a bit more tricky seeing as the column names and data are all over the place. We find the columns we need and create a dict storing their original names and the ones we want to rename them to and then we locate the said columns, rename them and change the zip code column so it has the same form as our original social capital dataframe's zip code column

[48]:
```
name_map = {
    "Geographic Area Name": "zip_code",
    "Estimate!!SEX AND AGE!!Total population" : "Population",
    "Estimate!!RACE!!Total population!!One race!!White": "White",
    "Estimate!!RACE!!Total population!!One race!!Black or African American":␣
 ↪"Black",
    "Estimate!!RACE!!Total population!!One race!!Asian": "Asian",
    "Estimate!!RACE!!Total population!!One race!!Some other race": "Other",
    "Estimate!!RACE!!Total population!!One race!!Native Hawaiian and Other␣
 ↪Pacific Islander": "Pacific",
    "Estimate!!RACE!!Total population!!One race!!American Indian and Alaska␣
 ↪Native": "Alaskan"
}

columns = name_map.keys()
neighborhooddata = neighborhooddata.loc[:, columns]
neighborhooddata.rename(columns=name_map, inplace=True)
neighborhooddata.zip_code = neighborhooddata.zip_code.apply(lambda x: x[6:]).
 ↪apply(int)
```

We then do the same thing as we did for colleges to calculate the HHI

[49]:
```
neighborhooddata['HHI'] = neighborhooddata.apply(lambda x: 1 -
                         (np.square(x.White/x.Population) + np.square(x.Black/x.
 ↪Population) +
                          np.square(x.Asian/x.Population) + np.square(x.Other/x.
 ↪Population) +
                          np.square(x.Pacific/x.Population) + np.square(x.
 ↪Alaskan/x.Population)), axis = 1)
```

```
C:\Users\Kharnifex\AppData\Local\Temp\ipykernel_3832\2758329206.py:2:
RuntimeWarning:

invalid value encountered in longlong_scalars
```

```
C:\Users\Kharnifex\AppData\Local\Temp\ipykernel_3832\2758329206.py:3:
RuntimeWarning:

invalid value encountered in longlong_scalars

C:\Users\Kharnifex\AppData\Local\Temp\ipykernel_3832\2758329206.py:4:
RuntimeWarning:

invalid value encountered in longlong_scalars
```

We choose the columns that we want to have in our final dataframe from each one and merge the two dataframes into one, then drop null values

[50]:
```
neighborhooddata = neighborhooddata.loc[:,['zip_code', 'HHI']]
neighborhood_df = neighborhood_df.loc[:,['zip', 'num_below_p50',␣
 ↪'bias_grp_mem_zip']]
neigh_df = neighborhooddata.merge(neighborhood_df, 'inner', left_on='zip_code',␣
 ↪right_on='zip')
neigh_df = neigh_df.dropna()
```

This is what the merged dataframe looks like

[51]:
```
neigh_df
```

[51]:
```
       zip_code       HHI    zip  num_below_p50  bias_grp_mem_zip
0          1001  0.143371   1001     995.787468           0.02434
1          1002  0.409802   1002    1312.117077           0.09856
3          1005  0.094008   1005     381.519745           0.00850
4          1007  0.101516   1007     915.396667          -0.01188
8          1013  0.295885   1013    2616.550354           0.13700
…           …         …     …             …                 …
23022     99840  0.249842  99840      75.419144          -0.08429
23023     99901  0.526235  99901    1192.299809           0.05710
23024     99921  0.512764  99921     365.768661           0.06010
23026     99926  0.430652  99926     311.014252           0.00877
23027     99929  0.507207  99929     313.282990           0.01350

[18329 rows x 5 columns]
```

We create bins (ventiles) for the dataframe and create a category column to store which bin each row belongs in

[52]:
```
bins2 = pd.cut(neigh_df.HHI, bins = 20, labels = func(np.arange(0,100,5)))
neigh_df['category'] = bins2
```

Then we calculate the original weighted hhi (we will need to divide it, again, same logic as with colleges) and do the same spaghetti as we did last time around for the group bys

[53]:
```
neigh_df['weighted_hhi'] = neigh_df.apply((lambda x: x.HHI * x.num_below_p50),
 ↪axis=1)

ndf_1 = neigh_df.groupby('category')\
               .sum(numeric_only=True).reset_index()\
               .loc[:,['category', 'num_below_p50']]

ndf_2 = neigh_df.groupby('category')\
            .mean(numeric_only=True).reset_index()\
            .loc[:,['category', 'bias_grp_mem_zip']]

ndf_3 = neigh_df.groupby('category')\
                .sum(numeric_only=True).reset_index()\
                .loc[:,['category', 'weighted_hhi']]

ndf_4 = ndf_2.merge(ndf_3, 'inner', left_on = 'category', right_on='category')

final_neigh_df = ndf_4.merge(ndf_1, 'inner', left_on = 'category',
 ↪right_on='category')
```

we then calculate the weighted HHI for each bin and multiply the friending bias by 100 to get percentage values

[54]:
```
final_neigh_df['weighted_hhi']= final_neigh_df.weighted_hhi / final_neigh_df.
 ↪num_below_p50
final_neigh_df['bias_grp_mem_zip'] = final_neigh_df['bias_grp_mem_zip'].
 ↪apply(lambda x: x*100)
```

Our final dataframe for neighborhoods ends up looking like this:

[55]:
```
final_neigh_df
```

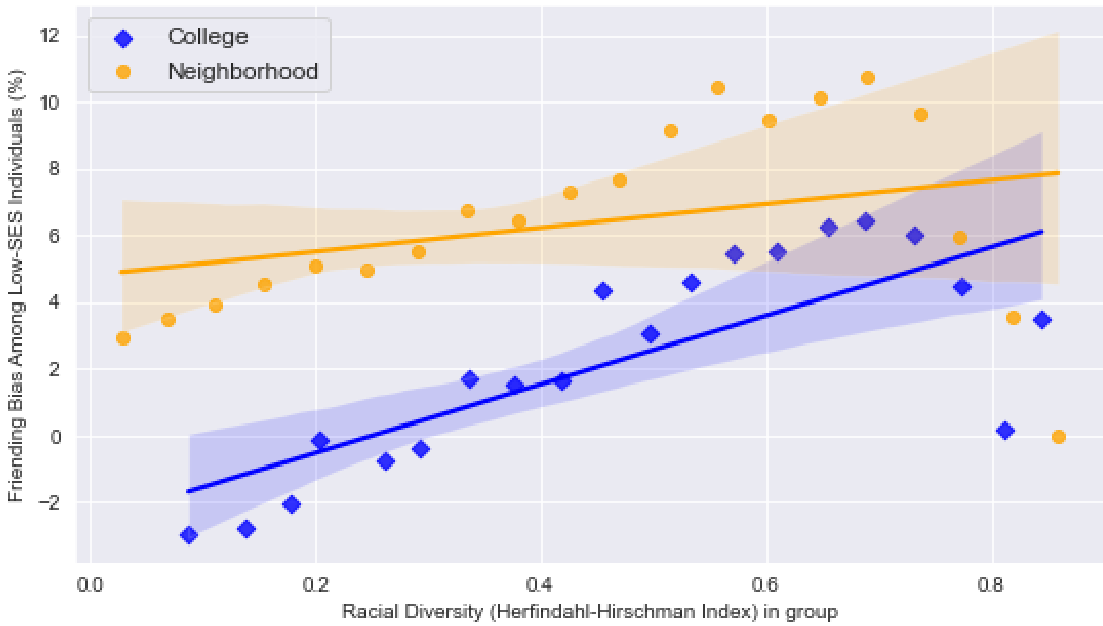
[55]:

	category	bias_grp_mem_zip	weighted_hhi	num_below_p50
0	0%	2.967081	0.027950	7.053599e+05
1	5%	3.496025	0.068683	1.883347e+06
2	10%	3.939225	0.110428	1.956676e+06
3	15%	4.515862	0.154960	1.883843e+06
4	20%	5.106557	0.200363	1.938751e+06
5	25%	4.963009	0.244507	1.817989e+06
6	30%	5.534455	0.290245	1.768873e+06
7	35%	6.769374	0.334976	1.934900e+06
8	40%	6.450838	0.380292	2.038970e+06
9	45%	7.286047	0.425425	2.205025e+06
10	50%	7.689960	0.469228	2.238459e+06
11	55%	9.145145	0.514137	2.539368e+06
12	60%	10.435390	0.556858	2.962135e+06
13	65%	9.441508	0.602812	2.368001e+06
14	70%	10.131747	0.647298	1.921820e+06

15	75%	10.731133	0.690222	1.452281e+06
16	80%	9.652726	0.735749	6.448763e+05
17	85%	5.975821	0.771507	1.610105e+05
18	90%	3.542077	0.818242	3.233971e+04
19	95%	0.003000	0.858684	1.010231e+04

Finally we can actually plot the two dataframes

[56]:
```
fig = plt.gcf()
fig.set_size_inches(9, 5)

sp5 = sns.regplot(x= 'weighted_hhi', y='bias_own_ses_college', data =␣
 ↪final_col_df, color = 'blue', label = 'College', marker="D")
sp5 = sns.regplot(x= 'weighted_hhi', y='bias_grp_mem_zip', data =␣
 ↪final_neigh_df, color = 'orange', label = 'Neighborhood' )

sp5.set(xlabel = 'Racial Diversity (Herfindahl-Hirschman Index) in group',
        ylabel = 'Friending Bias Among Low-SES Individuals (%)')

plt.legend(fontsize=12)
plt.show()
```



[ ]: