Data collection procedures

Dylan Weber

June 21, 2017

Things to note before you get started

- All the scripts will write their output files to the directory they are run in.
- You will need at least one reddit account that is the developer of a "script application." (i.e. an account with a "client ID" and "client secret"). See the next section on obtaining credentials from reddit.
- You will need Python. The scripts were written in version 2.7 and are therefore not guaranteed (and will probably not) to work with versions $\geq$ 3.
- You will need to install the "numpy" and "praw" packages to Python. If you're unsure how to do this google "installing python packages with pip."
- Ideally these scripts will be run in a linux environment. This is to take advantage of the "screen" command in order to achieve reasonable run-times on large sets of data. This will be more clear once you read the main implementation instructions. There may be work arounds to this in Windows that I'm not aware of. If you know of any please let me know!
- Please send all bugs/ideas for improvement to djw009@gmail.com

Obtaining script credentials from reddit

These scripts take advantage of the Python reddit API wrapper "PRAW". In order to access reddit's API, one needs credentials supplied from reddit.

1. Navigate to reddit.com and create an account.
2. Log into your account and navigate to the preferences tab found in the upper right hand corner of the screen.
3. Click the "apps" tab found in the upper left hand corner of the screen.
4. Click the "Are you a developer? create an app!" button. IF you already have apps click the "create another app" button.
5. Fill out the form, make sure you choose "script app' and NOT "web app" or "installed app".

6. Your client ID is the alpha-numeric string that appears under the name of your app after you've finished creating it. Your client secret is the alpha-numeric string that appears in the "client secret" field.

7. Ideally to run these scripts you should create several script apps to obtain several client ID and client secret pairs. Keep track of them somewhere.

Collecting interaction data from a subreddit

Step 1 - Generating username list

1. Make sure the scripts are in the directory that you want ouput files written to.

2. Run usercounter_final.py with the following command:

   ```
   python -i usercounter_final.py
   ```

   What does the script do?. *usercounter_final "watches" a subreddit of your choosing. It first creates an empty python dictionary called name_list ("the namelist"). Each time a submission is made to the subreddit you chose, usercounter grabs the author's name. If the author is not already in the name list it adds the author's username to the namelist with a submission counter value of one and a unique usernumber. The first user gets 0 the second user gets 1 and so on. After the scrape is completely finished an example entry of name_list would look like:*

   ```
   'username':[4,64]
   ```

   So, the user with the username 'username' was the 64th user collected by the script and made four submissions to the subreddit of interest while the script was running. The script also creates a .txt file for that user and stores the submission's ID in that .txt file. If the author IS in the namelist, usercounter adds one to their submission counter and adds the current submission ID to the .txt file for that user.

3. The script will prompt you to input what subreddit you'd like to scrape data from. You must type the name EXACTLY as it appears on reddit with no spaces before or after. If the script fails to run make sure you didn't accidentally input a space after typing the subreddit name.

4. The script will prompt you what you'd like to name the write file. I usually choose the name of the subreddit and the start time/date.

5. The script will prompt you for the name of the folder to store submission IDs. Inside this folder will be a .txt file named for each username collected during the scrape containing the IDs of the submissions made by that user during the scrape.

6. the script will then begin to run, indicated by its printing of messages such as:

   ```
   'username' is a new submitter
   ```

or:

```
'username''s␣count␣is␣n
```

7. Let the script run for as long as you are interested in collecting user data from your subreddit of choice. The script will write the namelist to a .csv file. End the script with Ctrl-C. At this point you can choose to continue the data collection process in your current session of Python; the namelist is stored to the variable:

   ```
   name_list
   ```

   Or, you can exit your current session and load the namelist from the .csv file written by usercounter to a variable of your choosing in a new session at any point later in the future (see next section).

Step 2 Generating the interaction data - the fast way

If you need to load namelist from the .csv file written by usercounter read the bullet points preceding step 1. Otherwise proceed to step 1.

- If you need to import the namelist from the .csv file written by user-counter first start python and import multi_breakdown with the following command:

  ```
  import multi_breakdown
  ```

- We are going to use the load_andconvert_namelists_todict function. If we want to name the variable that we load the namelist to "name_list" and the .csv file that the namelist was written to is called "namelist.csv" we call the function with the following command:

  ```
  name_list = multi_breakdown.load_andconvert_namelists_todict('namelist.csv
  ```

- the namelist should now be stored to a variable named name_list.

For the rest of this section I will assume you will be running the scripts in a linux environment, with the screen command available, have n clientIDs/client passwords, and that the namelist is stored to the variable name_list.

1. We first need to breakdown the namelist into n sub-namelists. We will do this using the dictbreaker function in the multi_breakdown module. First, import the multi_breakdown module with:

   ```
   import multi_breakdown
   ```

   What does the script do?. *dictbreaker takes a dictionary and an integer, i, as arguments. It outputs a tuple consisting of ten subdictionaries (this can be changed easily see appendix). It takes the first i entries of the given dictionary and adds them to the first subdictionary in the tuple, then it takes the next i entries and adds them to the second subdictionary in the tuple, and so on, until the given dictionary is empty. Therefore, if the integer is set high enough the output tuple may contain empty dictionaries. This is ok.*

NOTE:. *In its current form, dictbreaker does not support breaking a dictionary into more then 10 sub dictionaries. If you have more then 10 clientIDs you will want to do this. It is a relatively easy fix, but involves modifying the source code of dictbreaker. See the appendix.*

2. Then, check the length of your namelist with:

   ```
   len(name_list)
   ```

3. Divide the length of your namelist by n and round up (I will refer to this number as x), this is the maximum number of items in each sub dictionary.

4. call dictbreaker to break the namelist into subdictionaries with (the subdictionaries variable name is arbitrary):

   ```
   subdictionaries = multi_breakdown.dictbreaker(name_list, x)
   ```

5. Now, we need to write each subdictionary in the tuple subdictionaries to its own .csv file to be loaded to variables in different screen sessions later. To do this, we will call the listofdicts_writer from multi_breakdown with:

   ```
   listofdicts_writer(subdictionaries)
   ```

   The script will prompt you for how many nonempty dictionaries are in the tuple subdictionaries. It will then write a csv file for each subdictionary in the tuple and will name these files "j.csv" where j is an integer.

6. Quit your current session of Python. Make n screen sessions (one for each client ID you have, name them in a way you'll remember). Enter one of the sessions. In that session start python and import multi_breakdown again.

7. Now we need to generate the interaction data for one subdictionary. We do this by calling the bot_scrape() function from multi_breakdown with:

   ```
   interactiondata_j = bot_scrape()
   ```

   What does the script do?. *bot_scrape() will ask for a subdictionary of the name_list, the name_list and a set of reddit credentials. For each username in the subdictionary the script checks every other user in the name_list and counts how many times each user in the name_list has been a top level replier to a submission made by the current username or replied to a comment made by the current username. It then records that count as a value in a dictionary with the key (i,j) where i and j are the usernumbers of both users. The dictionary is written to a .csv file and returns as a variable. For example if user 45 was in a subdictionary that you gave to bot_scrape and the following item was in the returned dictionary:*

   ```
   (65, 34):8
   ```

   Then, that means that user 34 was a top-level replier on a submission or replied to a comment by user 65, 8 times.

8. The script will prompt you for your reddit username, password, a client ID, a client secret, and the names of the .csv files that the namelist and the subdictionary are written to. Input these EXACTLY with no spaces before or after. The script will then prompt you for a name for the output csv file (I will assume you named it interaction_data_j.csv). The script should begin to run, indicated by its printing of messages such as:

   ```
   'user_in_namelist' has been a toplevel
   replier to 'user_in_subdictionary' i times'
   ```

   or

   ```
   'user_in_namelist' has replied to a comment
   by 'user_in_subdictionary' i times'
   ```

   The script will return the resulting interaction data as a variable and write it to a .csv file with the name you supplied. At this point you can quit Python and leave this session of screen.

9. repeat steps 6-8 for all the subdictionaries. You should then have a interaction data csv file for each subdictionary.

10. Start python and import multi_breakdown. We need to load the interaction data for each subdictionary to a variable in python using the load_andconvert_namelists_todict function in multi_breakdown. Since this function name is annoyingly long and we're going to use it n times I'd suggest renaming it:

    ```
    load = multi_breakdown.load_andconvert_namelists_todict
    ```

    Now, for each interaction data csv file you generated enter the following:

    ```
    interaction_data_j = load('interaction_data_j.csv')
    ```

    You should now have n different variables each storing the interaction data generated by running bot_scrape on each subdictionary.

11. We now need to merge all these interaction data dictionaries into one interaction data dictionary for the whole scrape using tuplemerger (this is all tuplemerger does, it dosen't merit its own explanation) with the command:

    ```
    interaction_data_final = multi_breakdown.tuplemerger(interaction_data_1,
    interaction_data_2,..., interaction_data_n)
    ```

12. Finally, we need to import array_maker:

    ```
    import array_maker
    ```

    and use it to store the interaction data in a matrix format. This script has an option to symmetrize the matrix. If you enter 'yes' when it prompts you the ij$^{\text{th}}$ and ji$^{\text{th}}$ entries of the resulting matrix will be equal and will represent the number of interactions between user i and user j. If you enter 'no' the ij$^{\text{th}}$ entry of the matrix will represent how many times user j was a top-level comment to a submission by user i or replied to a comment by user i. The function is called with the command:

```
matrix = array_maker.array_maker(interaction_data_final)
```

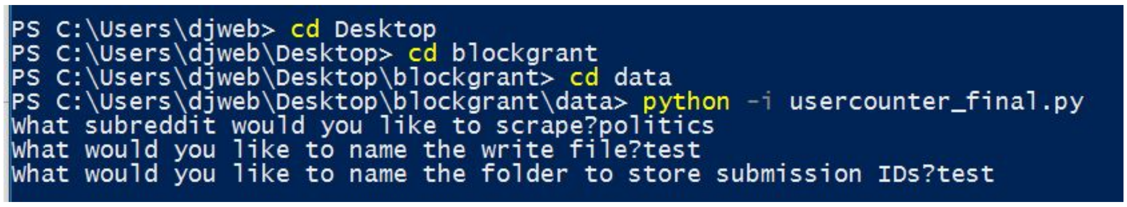
This function will return its output as a variable and write it to a .csv file that can be used import the matrix in any other supported language (MATLAB, R, etc.)

A sample run with pictures

The following set of pictures will show you sample inputs and outputs. However note that they were taken over many different scrapes so the outputs will not correspond to each other, just use them as a general guide. The names of the variables and their order match the instructions above.

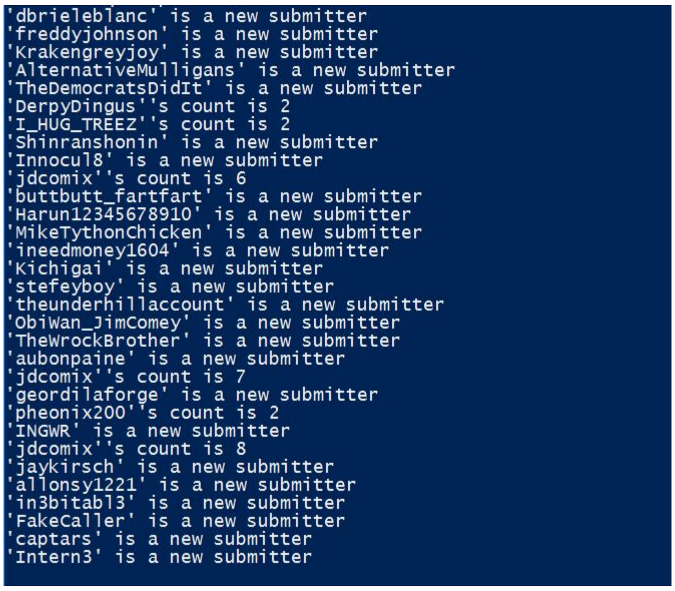
usercounter

input



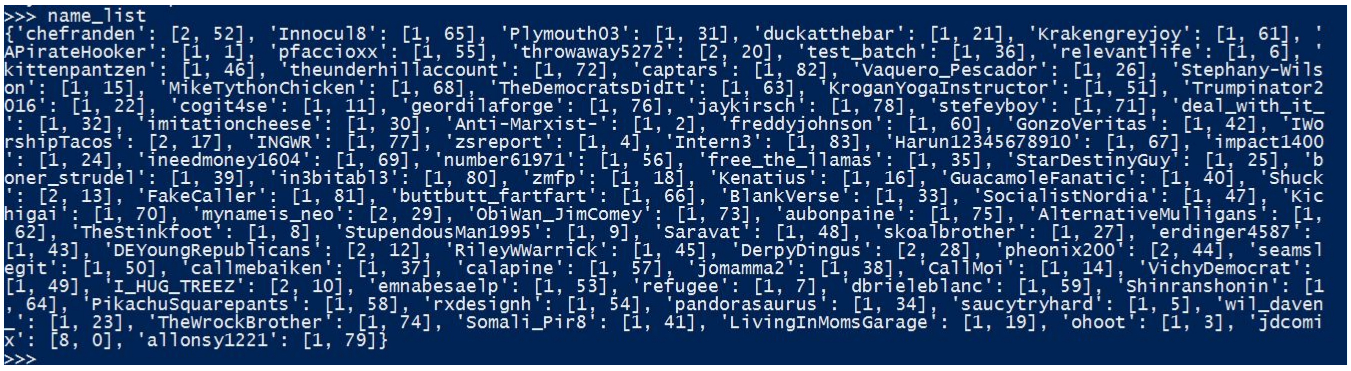
```
PS C:\Users\djweb> cd Desktop
PS C:\Users\djweb\Desktop> cd blockgrant
PS C:\Users\djweb\Desktop\blockgrant> cd data
PS C:\Users\djweb\Desktop\blockgrant\data> python -i usercounter_final.py
What subreddit would you like to scrape?politics
What would you like to name the write file?test
What would you like to name the folder to store submission IDs?test
```

running



```
'dbrieleblanc' is a new submitter
'freddyjohnson' is a new submitter
'Krakengreyjoy' is a new submitter
'AlternativeMulligans' is a new submitter
'TheDemocratsDidIt' is a new submitter
'DerpyDingus''s count is 2
'I_HUG_TREEZ''s count is 2
'Shinranshonin' is a new submitter
'Innocul8' is a new submitter
'jdcomix''s count is 6
'buttbutt_fartfart' is a new submitter
'Harun12345678910' is a new submitter
'MikeTythonChicken' is a new submitter
'ineedmoney1604' is a new submitter
'Kichigai' is a new submitter
'stefeyboy' is a new submitter
'theunderhillaccount' is a new submitter
'ObiWan_JimComey' is a new submitter
'TheWrockBrother' is a new submitter
'aubonpaine' is a new submitter
'jdcomix''s count is 7
'geordilaforge' is a new submitter
'pheonix200''s count is 2
'INGWR' is a new submitter
'jdcomix''s count is 8
'jaykirsch' is a new submitter
'allonsy1221' is a new submitter
'in3bitabl3' is a new submitter
'FakeCaller' is a new submitter
'captars' is a new submitter
'Intern3' is a new submitter
```

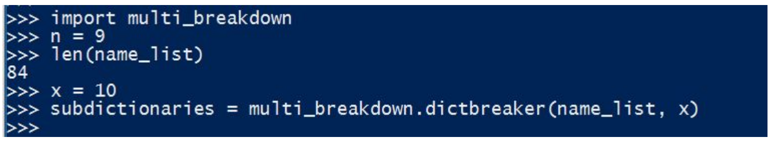
output



```
>>> name_list
{'chefranden': [2, 52], 'Innocul8': [1, 65], 'Plymouth03': [1, 31], 'duckatthebar': [1, 21], 'Krakengreyjoy': [1, 61], '
APirateHooker': [1, 1], 'pfaccioxx': [1, 55], 'throwaway5272': [2, 20], 'test_batch': [1, 36], 'relevantlife': [1, 6], '
kittenpantzen': [1, 46], 'theunderhillaccount': [1, 72], 'captars': [1, 82], 'Vaquero_Pescador': [1, 26], 'Stephany-Wils
on': [1, 15], 'MikeTythonChicken': [1, 68], 'TheDemocratsDidIt': [1, 63], 'KroganYogaInstructor': [1, 51], 'Trumpinator2
016': [1, 22], 'cogit4se': [1, 11], 'geordilaforge': [1, 76], 'jaykirsch': [1, 78], 'stefeyboy': [1, 71], 'deal_with_it_
': [1, 32], 'imitationcheese': [1, 30], 'Anti-Marxist-': [1, 2], 'freddyjohnson': [1, 60], 'GonzoVeritas': [1, 42], 'IWo
rshipTacos': [2, 17], 'INGWR': [1, 77], 'zsreport': [1, 4], 'Intern3': [1, 83], 'Harun12345678910': [1, 67], 'impact1400
': [1, 24], 'ineedmoney1604': [1, 69], 'number61971': [1, 56], 'free_the_llamas': [1, 35], 'StarDestinyGuy': [1, 25], 'b
oner_strudel': [1, 39], 'in3bitabl3': [1, 80], 'zmfp': [1, 18], 'Kenatius': [1, 16], 'GuacamoleFanatic': [1, 40], 'Shuck
': [2, 13], 'FakeCaller': [1, 81], 'buttbutt_fartfart': [1, 66], 'BlankVerse': [1, 33], 'SocialistNordia': [1, 47], 'Kic
higai': [1, 70], 'mynameis_neo': [2, 29], 'ObiWan_JimComey': [1, 73], 'aubonpaine': [1, 75], 'AlternativeMulligans': [1,
 62], 'TheStinkfoot': [1, 8], 'StupendousMan1995': [1, 9], 'Saravat': [1, 48], 'skoalbrother': [1, 27], 'erdinger4587':
[1, 43], 'DEYoungRepublicans': [2, 12], 'RileyWWarrick': [1, 45], 'DerpyDingus': [2, 28], 'pheonix200': [2, 44], 'seamsl
egit': [1, 50], 'callmebaiken': [1, 37], 'calapine': [1, 57], 'jomamma2': [1, 38], 'CallMoi': [1, 14], 'VichyDemocrat':
[1, 49], 'I_HUG_TREEZ': [2, 10], 'emnabesaelp': [1, 53], 'refugee': [1, 7], 'dbrieleblanc': [1, 59], 'Shinranshonin': [1
, 64], 'PikachuSquarepants': [1, 58], 'rxdesignh': [1, 54], 'pandorasaurus': [1, 34], 'saucytryhard': [1, 5], 'wil_daven
_': [1, 23], 'TheWrockBrother': [1, 74], 'Somali_Pir8': [1, 41], 'LivingInMomsGarage': [1, 19], 'ohoot': [1, 3], 'jdcomi
x': [8, 0], 'allonsy1221': [1, 79]}
>>>
```

dictbreaker

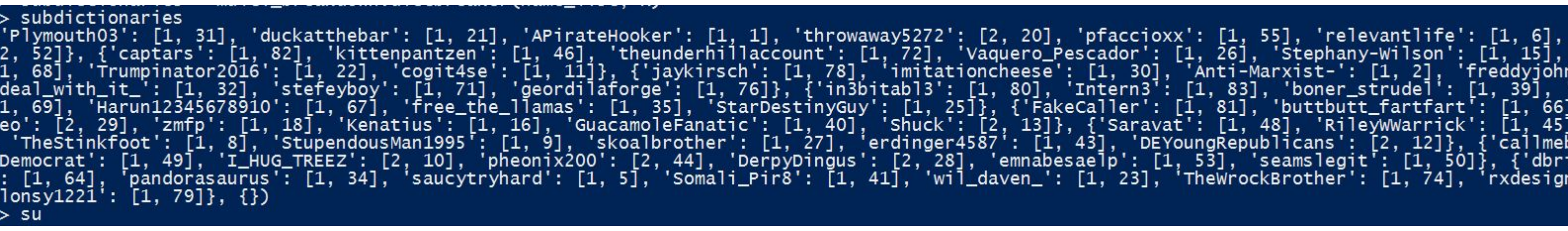
input



```
>>> import multi_breakdown
>>> n = 9
>>> len(name_list)
84
>>> x = 10
>>> subdictionaries = multi_breakdown.dictbreaker(name_list, x)
>>>
```

output

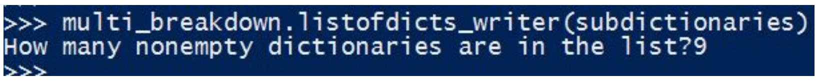
Figure 1: Notice the empty dictionary in the last entry of the subdictionaries tuple output by dictbreaker



```
> subdictionaries
'Plymouth03': [1, 31], 'duckatthebar': [1, 21], 'APirateHooker': [1, 1], 'throwaway5272': [2, 20], 'pfaccioxx': [1, 55], 'relevantlife': [1, 6],
2, 52]}, {'captars': [1, 82], 'kittenpantzen': [1, 46], 'theunderhillaccount': [1, 72], 'Vaquero_Pescador': [1, 26], 'Stephany-Wilson': [1, 15],
1, 68], 'Trumpinator2016': [1, 22], 'cogit4se': [1, 11]}, {'jaykirsch': [1, 78], 'imitationcheese': [1, 30], 'Anti-Marxist-': [1, 2], 'freddyjohn
deal_with_it_': [1, 32], 'stefeyboy': [1, 71], 'geordilaforge': [1, 76]}, {'in3bitabl3': [1, 80], 'Intern3': [1, 83], 'boner_strudel': [1, 39],
1, 69], 'Harun12345678910': [1, 67], 'free_the_llamas': [1, 35], 'StarDestinyGuy': [1, 25]}, {'FakeCaller': [1, 81], 'buttbutt_fartfart': [1, 66
eo': [2, 29], 'zmfp': [1, 18], 'Kenatius': [1, 16], 'GuacamoleFanatic': [1, 40], 'Shuck': [2, 13]}, {'Saravat': [1, 48], 'RileyWWarrick': [1, 45
 'TheStinkfoot': [1, 8], 'StupendousMan1995': [1, 9], 'skoalbrother': [1, 27], 'erdinger4587': [1, 43], 'DEYoungRepublicans': [2, 12]}, {'callme
Democrat': [1, 49], 'I_HUG_TREEZ': [2, 10], 'pheonix200': [2, 44], 'DerpyDingus': [2, 28], 'emnabesaelp': [1, 53], 'seamslegit': [1, 50]}, {'dbr
: [1, 64], 'pandorasaurus': [1, 34], 'saucytryhard': [1, 5], 'Somali_Pir8': [1, 41], 'wil_daven_': [1, 23], 'TheWrockBrother': [1, 74], 'rxdesig
lonsy1221': [1, 79]}, {})
> su
```

listofdicts_writer

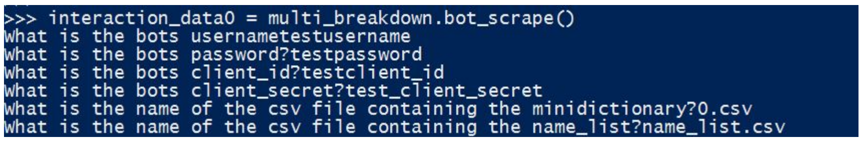
input



```
>>> multi_breakdown.listofdicts_writer(subdictionaries)
How many nonempty dictionaries are in the list?9
>>>
```

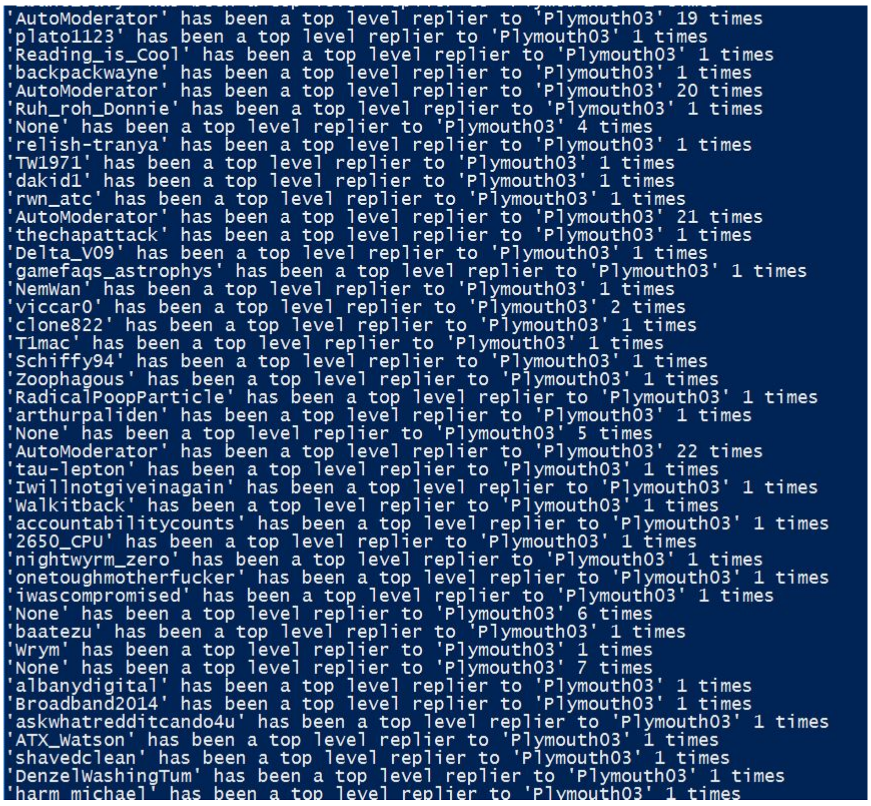
bot_scrape

input



```
>>> interaction_data0 = multi_breakdown.bot_scrape()
What is the bots usernametestusername
What is the bots password?testpassword
What is the bots client_id?testclient_id
What is the bots client_secret?test_client_secret
What is the name of the csv file containing the minidictionary?0.csv
What is the name of the csv file containing the name_list?name_list.csv
```

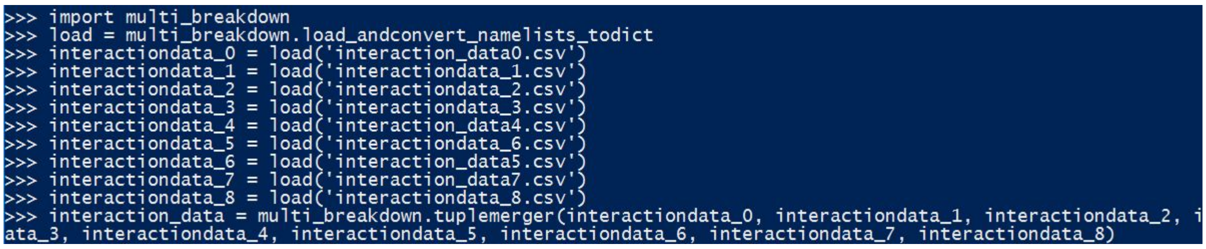
running



```
'AutoModerator' has been a top level replier to 'Plymouth03' 19 times
'plato1123' has been a top level replier to 'Plymouth03' 1 times
'Reading_is_Cool' has been a top level replier to 'Plymouth03' 1 times
'backpackwayne' has been a top level replier to 'Plymouth03' 1 times
'AutoModerator' has been a top level replier to 'Plymouth03' 20 times
'Ruh_roh_Donnie' has been a top level replier to 'Plymouth03' 1 times
'None' has been a top level replier to 'Plymouth03' 4 times
'relish-tranya' has been a top level replier to 'Plymouth03' 1 times
'TW1971' has been a top level replier to 'Plymouth03' 1 times
'dakid1' has been a top level replier to 'Plymouth03' 1 times
'rwn_atc' has been a top level replier to 'Plymouth03' 1 times
'AutoModerator' has been a top level replier to 'Plymouth03' 21 times
'thechapattack' has been a top level replier to 'Plymouth03' 1 times
'Delta_V09' has been a top level replier to 'Plymouth03' 1 times
'gamefaqs_astrophys' has been a top level replier to 'Plymouth03' 1 times
'NemWan' has been a top level replier to 'Plymouth03' 1 times
'viccar0' has been a top level replier to 'Plymouth03' 2 times
'clone822' has been a top level replier to 'Plymouth03' 1 times
'T1mac' has been a top level replier to 'Plymouth03' 1 times
'Schiffy94' has been a top level replier to 'Plymouth03' 1 times
'Zoophagous' has been a top level replier to 'Plymouth03' 1 times
'RadicalPoopParticle' has been a top level replier to 'Plymouth03' 1 times
'arthurpaliden' has been a top level replier to 'Plymouth03' 1 times
'None' has been a top level replier to 'Plymouth03' 5 times
'AutoModerator' has been a top level replier to 'Plymouth03' 22 times
'tau-lepton' has been a top level replier to 'Plymouth03' 1 times
'Iwillnotgiveinagain' has been a top level replier to 'Plymouth03' 1 times
'Walkitback' has been a top level replier to 'Plymouth03' 1 times
'accountabilitycounts' has been a top level replier to 'Plymouth03' 1 times
'2650_CPU' has been a top level replier to 'Plymouth03' 1 times
'nightwyrm_zero' has been a top level replier to 'Plymouth03' 1 times
'onetoughmotherfucker' has been a top level replier to 'Plymouth03' 1 times
'iwascompromised' has been a top level replier to 'Plymouth03' 1 times
'None' has been a top level replier to 'Plymouth03' 6 times
'baatezu' has been a top level replier to 'Plymouth03' 1 times
'Wrym' has been a top level replier to 'Plymouth03' 1 times
'None' has been a top level replier to 'Plymouth03' 7 times
'albanydigital' has been a top level replier to 'Plymouth03' 1 times
'Broadband2014' has been a top level replier to 'Plymouth03' 1 times
'askwhatredditcando4u' has been a top level replier to 'Plymouth03' 1 times
'ATX_Watson' has been a top level replier to 'Plymouth03' 1 times
'shavedclean' has been a top level replier to 'Plymouth03' 1 times
'DenzelWashingTum' has been a top level replier to 'Plymouth03' 1 times
'harm_michael' has been a top level replier to 'Plymouth03' 1 times
```

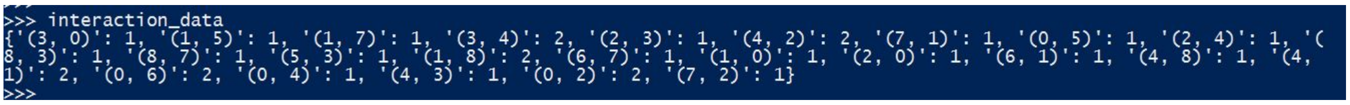
tuplemerger

input



```
>>> import multi_breakdown
>>> load = multi_breakdown.load_andconvert_namelists_todict
>>> interactiondata_0 = load('interaction_data0.csv')
>>> interactiondata_1 = load('interactiondata_1.csv')
>>> interactiondata_2 = load('interactiondata_2.csv')
>>> interactiondata_3 = load('interactiondata_3.csv')
>>> interactiondata_4 = load('interaction_data4.csv')
>>> interactiondata_5 = load('interactiondata_6.csv')
>>> interactiondata_6 = load('interaction_data5.csv')
>>> interactiondata_7 = load('interaction_data7.csv')
>>> interactiondata_8 = load('interactiondata_8.csv')
>>> interaction_data = multi_breakdown.tuplemerger(interactiondata_0, interactiondata_1, interactiondata_2, i
ata_3, interactiondata_4, interactiondata_5, interactiondata_6, interactiondata_7, interactiondata_8)
```

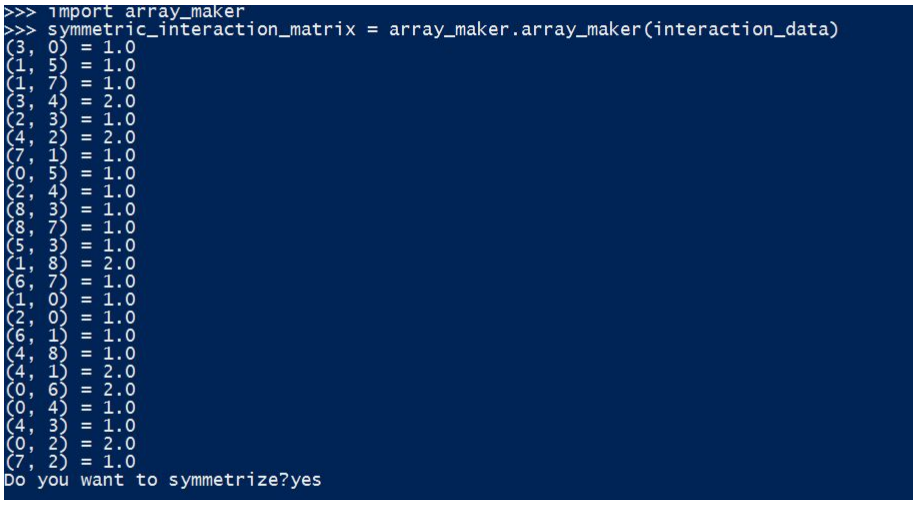
output



```
>>> interaction_data
{'(3, 0)': 1, '(1, 5)': 1, '(1, 7)': 1, '(3, 4)': 2, '(2, 3)': 1, '(4, 2)': 2, '(7, 1)': 1, '(0, 5)': 1, '(2, 4)': 1, '(
8, 3)': 1, '(8, 7)': 1, '(5, 3)': 1, '(1, 8)': 2, '(6, 7)': 1, '(1, 0)': 1, '(2, 0)': 1, '(6, 1)': 1, '(4, 8)': 1, '(4,
1)': 2, '(0, 6)': 2, '(0, 4)': 1, '(4, 3)': 1, '(0, 2)': 2, '(7, 2)': 1}
>>>
```

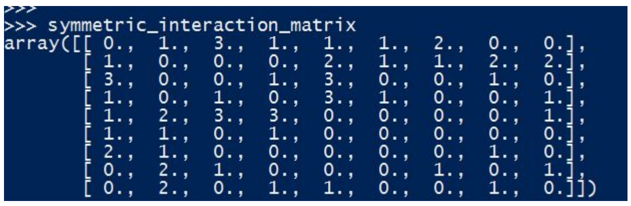
array _maker

input



```
>>> import array_maker
>>> symmetric_interaction_matrix = array_maker.array_maker(interaction_data)
(3, 0) = 1.0
(1, 5) = 1.0
(1, 7) = 1.0
(3, 4) = 2.0
(2, 3) = 1.0
(4, 2) = 2.0
(7, 1) = 1.0
(0, 5) = 1.0
(2, 4) = 1.0
(8, 3) = 1.0
(8, 7) = 1.0
(5, 3) = 1.0
(1, 8) = 2.0
(6, 7) = 1.0
(1, 0) = 1.0
(2, 0) = 1.0
(6, 1) = 1.0
(4, 8) = 1.0
(4, 1) = 2.0
(0, 6) = 2.0
(0, 4) = 1.0
(4, 3) = 1.0
(0, 2) = 2.0
(7, 2) = 1.0
Do you want to symmetrize?yes
```

output



```
>>>
>>> symmetric_interaction_matrix
array([[ 0.,  1.,  3.,  1.,  1.,  1.,  2.,  0.,  0.],
       [ 1.,  0.,  0.,  0.,  2.,  1.,  1.,  2.,  2.],
       [ 3.,  0.,  0.,  1.,  3.,  0.,  0.,  1.,  0.],
       [ 1.,  0.,  1.,  0.,  3.,  1.,  0.,  0.,  1.],
       [ 1.,  2.,  3.,  3.,  0.,  0.,  0.,  0.,  1.],
       [ 1.,  1.,  0.,  1.,  0.,  0.,  0.,  0.,  0.],
       [ 2.,  1.,  0.,  0.,  0.,  0.,  0.,  1.,  0.],
       [ 0.,  2.,  1.,  0.,  0.,  0.,  1.,  0.,  1.],
       [ 0.,  2.,  0.,  1.,  1.,  0.,  0.,  1.,  0.]])
```