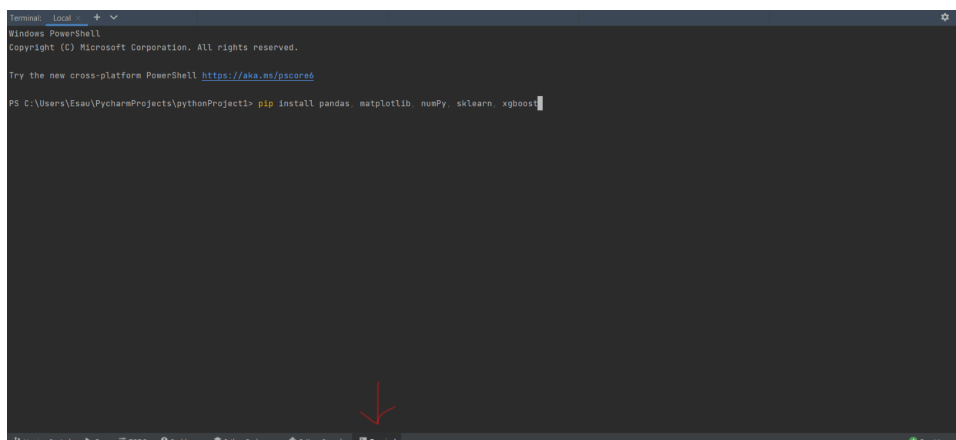


Installing Python Packages (Inside Pycharms)

1. We will be installing various python packages that are utilized later on in this project for things such as visualizations and analyzation of data
2. The packages we will be installing include:
 - a. pandas
 - b. matplotlib
 - c. numpy
 - d. sklearn
 - e. xgboost
3. Open your created Pycharms project and on the bottom there should be a tab on the very right named terminal. We will be utilizing pip to install the packages listed above.
4. Pip a package manager is already included when you downloaded Python 3.9 and Pycharms.
5. Inside the terminal type in the phrase: pip install pandas, matplotlib, numPy, sklearn, xgboost. Make sure this is typed out all in one line and pip will install the packages for use in your virtual environment
6. To check if you have all the packages installed look for the tab named Python packages at the bottom of the Pycharms window
7. Click the tab and look up each of the 5 packages to see if it was successfully installed
8. Another package named pickle that will be used later in the project is already included in Python 3.9 and does not need to be manually installed
9. It will take a while until all the packages are done and installed
10. You are now finished!



```
Terminal: Local
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

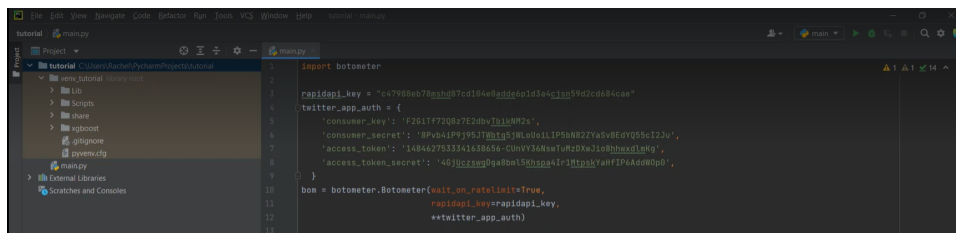
PS C:\Users\Esau\PycharmProjects\pythonProject1> pip install pandas matplotlib numpy sklearn xgboost
```

10. Gathering Data

Once you are in your PyCharm environment load up a new project to start entering in data. At the top of your project enter the lines

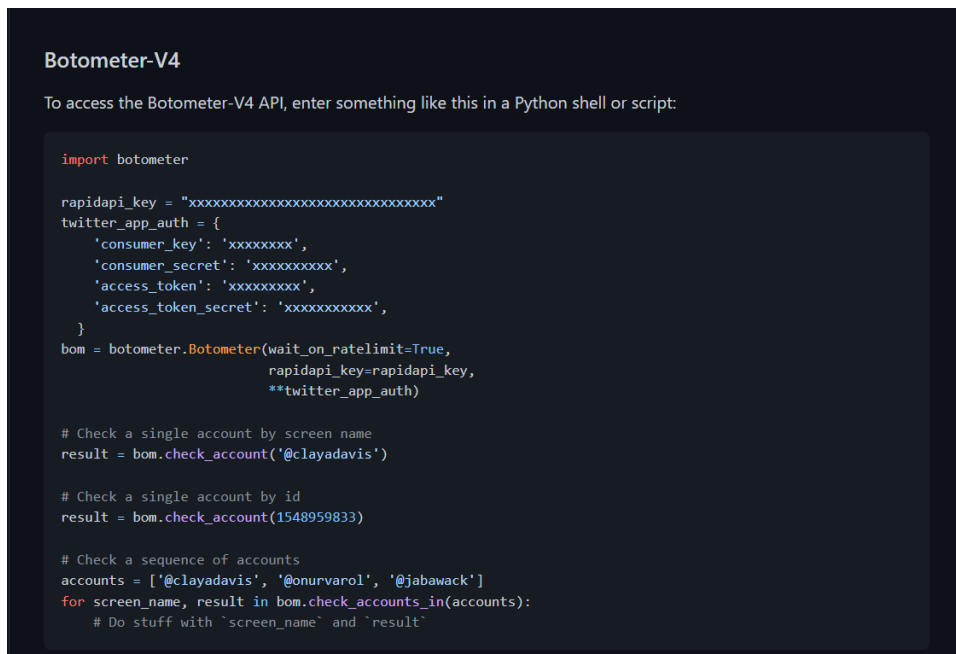
- Import botometer
- Import pprint

If you are using .env files to receive data you need to also type “import load_dotenv” at the top of the program. Next type in these lines of code into your program as we are going to have to utilize your newly created twitter developer account and botometer account in order to analyze and receive data.



```
1 import botometer
2
3 rapidapi_key = "c47988eb7b9b067cd184e8b0e4e1d34cc159d2c064cae"
4
5 twitter_app_auth = {
6     'consumer_key': '72011720847200vT11gW2s',
7     'consumer_secret': '1Pp041P9393710u1gJ9Lm0e1IP0a022Yabv8L0Y0f6c123u',
8     'access_token': '14040270331638654-D0y728AseT0R0Dta110RphaxD1a9',
9     'access_token_secret': '40J1C100Pq48w150h00461r1H0ssYahFIP6Adm0p0',
10 }
11
12 bom = botometer.Botometer(wait_on_ratelimit=True,
13                           rapidapi_key=rapidapi_key,
14                           **twitter_app_auth)
```

You can go to <https://github.com/IUNetSci/botometer-python> and go over the README which details how to use botomer with python.



```
Botometer-V4
To access the Botometer-V4 API, enter something like this in a Python shell or script:

import botometer

rapidapi_key = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
twitter_app_auth = {
    'consumer_key': 'xxxxxxx',
    'consumer_secret': 'xxxxxxxxxx',
    'access_token': 'xxxxxxx',
    'access_token_secret': 'xxxxxxxxxx',
}
bom = botometer.Botometer(wait_on_ratelimit=True,
                          rapidapi_key=rapidapi_key,
                          **twitter_app_auth)

# Check a single account by screen name
result = bom.check_account('@clayadavis')

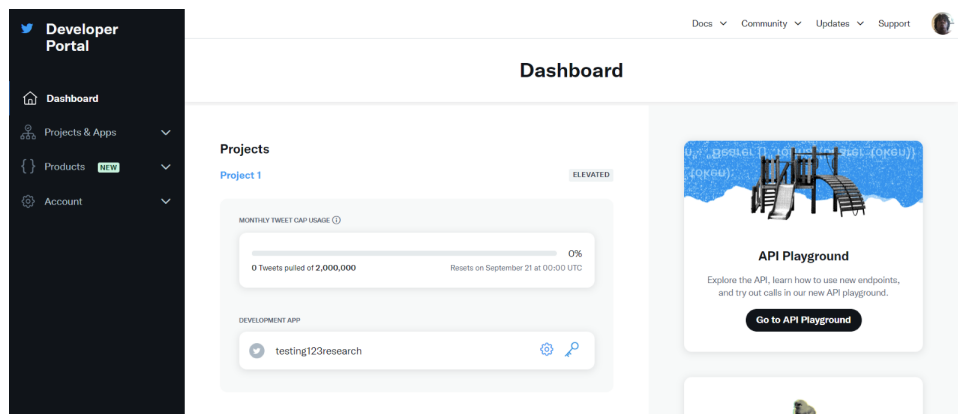
# Check a single account by id
result = bom.check_account(1548959833)

# Check a sequence of accounts
accounts = ['@clayadavis', '@onurvarol', '@jabawack']
for screen_name, result in bom.check_accounts_in(accounts):
    # Do stuff with `screen_name` and `result`
```

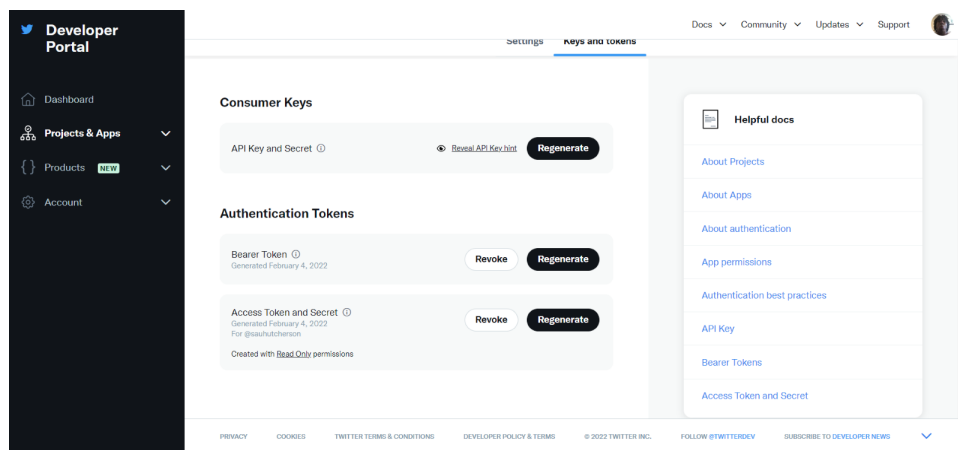
The values for “rapidapi key”, “consumer_key”, “consumer_secret”, “access_token”, “access_token_secret” need to be filled filled in with your own twitter developer account values as

mentioned above in the twitter developer section. Make sure to correctly input all the values into the correct spot or your code will not be able to run!

If you forgot your key head over to <https://developer.twitter.com/en/portal/dashboard> and sign in. Create a project and an app if you haven't already and then click the key next to the app you created for mine its testing123



After you click the key you should see all the necessary keys and you should be able to click regenerate to generate a new key for use.



Now you should be able to get data on individual twitter accounts. You can check an individual account and print the result utilizing this code.

```
13
14 # Check a single account by screen name
15 result = bom.check_account('@clayadavis')
16
17 # Check a single account by id
18 result = bom.check_account(1548959833)
```

The first line saves the data of the account you input into the variable result and the second line prints the entire data structure in an easy to read format.

Analyzing one account is fun but you can do more however through use of python we can analyze hundreds or thousands of accounts at once. You can do this by inputting your twitter ids into a list and then utilizing a for loop as shown below.

```
20 # Check a sequence of accounts
21 accounts = ['@clayadavis', '@onurvarol', '@jabawack']
22 for screen_name, result in bom.check_accounts_in(accounts):
23     print(screen_name)
24     print(result)
25 # Do stuff with 'screen_name' and 'result'
```

When this code is executed it will print out all of the twitter ids you have inputted into your list along with their data fields. Down below is a breakdown of all the data fields and what they mean. (Replace the print statement with a pprint statement for a nicer looking output!). From now on we will be using pprint over the usual print statement so make sure to use pprint.

- user: Twitter user object (from the user) plus the language inferred from majority of tweets
- raw scores: bot score in the [0,1] range, both using English (all features) and Universal (language-independent) features; in each case we have the overall score and the sub-scores for each bot class (see below for subclass names and definitions)
- display scores: same as raw scores, but in the [0,5] range
- cap: conditional probability that accounts with a score equal to or greater than this are automated; based on inferred language

Your Final code for this section should look like [this](#):

```
1 import botometer
2 from pprint import pprint
3
4 rapidapi_key = 'adba30d77bmsdcbfc478e5ccc58p18b171jsn5462eff7cbb'
5 twitter_app_auth = {
6     'consumer_key': 'w8CyF1Z1qyFZISDF1UnR97U4n',
7     'consumer_secret': 'MJFsEhID9wnYA3CCf0RbDime4jyA2tZXRW8InCdV08teren72h',
8 }
9 bom = botometer.Botometer(wait_on_ratelimit=True,
10                            rapidapi_key=rapidapi_key,
11                            **twitter_app_auth)
12
13 accounts = ['@clayadavis', '@onurvarol', '@jabawack']
14
15 for id, result in bom.check_accounts_in(accounts):
16     pprint(result)
```

Bot types scores field meanings are described below:

- fake_follower: bots purchased to increase follower counts
- self_declared: bots from botwiki.org
- Astroturf: manually labeled political bots and accounts involved in follow trains that systematically delete content
- spammer: accounts labeled as spambots from several datasets
- financial: bots that post using Cash tags
- other: miscellaneous other bots obtained from manual annotation, user feedback, etc.

All the information can be seen on the RapidAPI website: <https://rapidapi.com/OSoMe/api/botometer-pro/details>

