MixRank Assignment

About Approach

Given list holds 1000 unique websites. The script visits the csv and :

- loads the sites into a list for further filtering
- pings the items in the list `companies` and categorizes them based on their response code.

  Since there will be consequent requests, filtering the non-responsive websites gains time.
- As the second step, script goes through the filtered list `connected` and searches for favicons in 2 different places : `item/favicon.ico` and website's head.
- registers the findings into a new list.
- As the third step, the script visits the same `connected` and looks for meta:og-image or embedded png, svg files.
- I created a variable called barebones. For each `http://company.com`, barebones is `Company`. Because websites tend to use their name.capitalize() for logo naming.
- Finally, instead of using a for loop, I used concurrency. In the benchmark tests, provided 1500% more efficiency.

Encountered Difficulties

1. Websites use lots of external css files, there is no one way for all websites that can identify and target the css used for actually styling the website.
2. Even though we could , there are numerous background images and websites are not always friendly with their naming conventions for classes and ids, makes it more complicated.
3. some websites, including Mixrank, uses hardcoded styled strings to constitute their logos. This renders them to be unscrappable.
4. There are http vs https preferences of the websites, but tests yield that http:// convention is more fruitful thanks to the redirection. Still cause data loss.
5. some sites override the cookies and the return text is not giving actual body of the site. I used Session() but did not help in some cases.
6. Code does run normally on local environment, but when made derivation, only favicon function works whereas logo function returns none. Thus, I opted to ship code without using package manager. I would like this to be a discussion point

Problems With the current codebase:

Current codebase yields some false positives. Some logo URLs are not necessarily correct. This is mainly because of the structural weaknesses due to the difficulties encountered.

Script checks status.code. Sometimes status code returns 200 however the website content does not return favicon.ico but a error page. These are small amount of false positives, though.

Script makes several connections to the same website for different purposes. This negatively impacts the script running time and also increments the ping number, which may cause banishment in some cases.

Preferences and Reasons:

I gave timeout:2 seconds based on the severals tests made with 55 websites

> 1 second timeout resulted with 44 connections and took 55 seconds

> **2** second timeout resulted with **49** connections and took **70 seconds**

> 4 second timeout resulted with 49 connections and took 78 second

> 10 second timeout resulted with 49 connections and took 81 secons.

Thus, 2 seconds timeout is the sweet-spot between timing and performance.

I used BeautifulSoup with Requests although:

> Scrapy is faster, has built-in non-blocking mechanism, uses less resources and extendable.

I used BeautifulSoup with Requests because:

> It has less steep learning curve, can also use concurrency, easier to debug and maintain for smaller projects.

Tradeoff was resource + speed vs complexity. Given the situation, I selected bs4 over scrapy.

I created different output csv files.

The program creates 3 csv files: output.csv , favicon.csv, and non-connections.csv

I created non-connetions barely in order to test http vs https checks. Testing against failed connections gives better understanding whether connection failure is due to request protocol or possible other reason.

Although I could not come up with a solid logic regarding whether to write favicon url into the same csv file with logo url, I opted for writing them in separate file due to the bug I encountered while writing to the same file.

Bug was causing some shuffle in the order of list rows.

Benchmark Results with 1000 Websites

```
time python3 submit.py

265MB Ram Usage
```

```
max workers = 100

there was 1000 items in the list  and 829 of them are connected

loss of overall url list is 17.1%

amount of non-connections is 171 and amount of connections is 829

number of logos found:  468

found favicons: 822

python3 submit.py  166.74s user 58.37s system 91% cpu 4:05.53 total
```