Project Milestone 2
Tests

Michał Dyczko
Jakub Kała
Maciej Pawłowski
Krzysztof Spaliński

April 2020

This report is written to review data collecting and data preprocessing tests', which were conducted in order to examine scrapers' performance with initial data preprocessing. The document is divided in four separate sections, each summarizing actions performed within a distinct data source.

Each source was examined for their accessibility, reliability and durability. To achieve such results three main components were reviewed:

- object components - confirmation that expected data is extracted
- data management - confirmation that collected data is saved and stored in desired forms
- data accessibility - verification if the source is accessible and conditions in which it is possible

1 imgur

Images originated from imgur.com are accessible via API. Official Python library used to interact with API is deprecated, therefore it was necessary to code custom implementation of imgur API Client. To receive that, *requests* library were used. Unit tests validity of imgur scraper inspect 5 main aspects:

1. *test_obtain_data_from_imgur_returns_data_in_correct_timestamp_window* testing if obtained images are indeed in a time window described by 2 variables: *start_timestamp* and *end_timestamp*.
2. *test_obtain_data_from_imgur_downloads_images* testing whether obtained images are saved in a right directory.
3. *test_obtain_data_from_imgur_creates_json_file* testing if JSON file is saved under correct path.

4. *test_obtain_data_from_imgur_creates_log* testing if log file is saved under a correct path.

5. *test_obtain_data_from_imgur_returns_data_in_correct_format* testing whether returned JSON containing data about images is in a correct format.

As one can see in a log below, all of these tests are passed with a success:

```
[******* imgur](master)$ python3 tests.py
.....
----------------------------------------------------------------------
Ran 5 tests in 58.357s

OK
```

2 memedroid

Memes from memedroid.com were collected using the Python tool - BeautifulSoup. Each scraped object was containing desired pieces of information: an url, a title, the time at which the meme was uploaded, an upvote ratio representing meme's popularity and tags if applicable.

The performance of *memedroid_scraper.py* and whether it extracts all the desired information was examined by the *test_memedroid_scraper.py* module. The latter script contains 10 tests, each inspecting different aspects. To examine the scraper two html files (representing scraped objects) were created. Obtained results confirmed scraper's performance with respect to:

- meme content - each desired information was extracted in appropriate form, that is: the title, the url, the upvote ratio, the time of uploading and tags.
- meme compliance - video is expected to be omitted as well as if the examined object is of an appropriate format
- website accessibility - memedroid.com should be approachable and should provide such data: memes information and urls of following websites

The script *memedroid_usecase.py* examines data management. With the help of this file two objects were created: *memedroid_2020041911.json* and *memedroid_log_2020041911.txt*. The first one contains data uploaded to memedroid.com between 20.00 and 23.00 on 18.04.2020. It consists of 43 elements, each containing desired fields: the url, additional data and the filename. The latter file contains information of the data collected and the time at which it was performed. Both objects (.json and .txt) are containing expected information and are of desired forms.

To summarize, performed functionality tests confirmed *memedroid_scraper.py* high performance. With its help memes can be extracted and stored in desired forms.

3 Reddit

Memes from Reddit are scraped using well tested and documented library Praw which is using official Reddit API. Hence tests only checked, that the program does not fail in event of not valid URL and that files were created correctly.

4 Twitter

Memes from Twitter are being collected using official TwitterAPI via Python library Tweepy. Since we use official API and well tested and documented, tests are limited to ensure that output files have proper format.