FIT2107 Assignment 3 - Individual Report

Matthew James Sturgeon, mjstu3, 27833380

What bugs, if any, did you find with your unit tests? Please be brief.

Several minor bugs occurred when testing the analyse_tweets method. They were easy to identify because the failed test would be outlined, including which line had an error. The first bug we identified was with the Twitter retrieval limits (TweepError, status code 429), which occurred when too many calls to the API were made in a specific time frame. We solved this by using multiple tokens to allow for more frequent testing. Another bug we found was when validating the correct output of word frequencies, the test passed in Pycharm but failed in GitLab.

Did you think your testing strategy resulted in adequate tests? Please explain why or why not.

The majority of test cases for the analyse_tweets method were derived using a branch/statement coverage strategy. This allowed us to ensure that all potential outcomes of the algorithm were tested, depending on the input parameters. Even though our tests for this part achieved complete statement coverage, I believe that there were several other test cases which could have been included to result in adequate testing. For example, we did not include test cases for both date constraints and the minimum word count parameter because the coverage report showed that we had already achieved complete statement coverage. Based on this strategy, we would not have had an explicable reason for including these test cases, besides the fact that they are inputs which can be validated through testing. Despite this, I still believe that our program is supported by a sufficient number of test cases, which proved to cover all statements of the algorithm. Our testing strategy resulted in a high quality and suitable number of tests which assisted us with identifying important bugs in our program.

Do you think it works better for the programmer responsible for code to write the unit tests for it, or for somebody else to do it? Explain why.

The person who writes the code will be better suited to writing the test cases for it, because already have a good understanding of its functionality. Giving the task of testing to someone who has not written the code may result in lower quality test cases, because they need to take time to understand it before writing test cases. It depends on how complicated the algorithms are because for a simple program it would be easy for either of the writer of the code or another programmer to test it. However, for a large-scale, complicated program, it would be faster for the author of the code to test because they already have a good understanding of their work. For this reason, I think it works better for the programmer who wrote the code to perform unit testing.

How long did writing mock code take? Was it a major component of the overall effort?

Mocking took a good part of a day to complete, we were required to undertake research about mocking objects because neither of us had done it before. I had already written the test cases which called the method without mocking, thus making calls to the API, so it was quite simple to implement once we developed an understanding. It was a large component of the overall effort because we needed to determine the correct format of the object returned by Twitter. The mocked object had to imitate the object returned from the API, and consist of the status text and creation date.

What would you do differently next time if faced with a similar task?

It would have been helpful to know more about mocking before doing this task. We had not practiced it before so it was challenging to implement. I would also like to have done more research into how the Tweepy API works including how the access tokens work. Doing more study about testing strategies before conducting the tasks would have allowed me to save time during the actual task.