CSC110 Fall 2022 Assignment 3: Loops, Mutation, and Applications

Vrinda Subhash

November 1, 2022

Part 1: Data Analysis with Toronto Health

Complete this part in the provided `a3_part1.py` file. Do **not** include your solutions in this file.

Part 2: Loops and Mutation Debugging Exercise

1. test_star_wars PASSED
   test_legally_blonde FAILED
   test_transformers FAILED

2. test_legally_blonde failed because in the clean_text function, it was not properly converting the text to be lowercase. The code that was written to make the text lowercase just called str.lower on text, which returns a string, but that string was not assigned to anything and since str.lower does not mutate, the text was not changed to be lowercase. So, I changed the code to use variable reassignment to assign str.lower(text) to text so now text is the lowercase version. In the legally blonde review, a couple of the lexicon keywords started with an uppercase letter, and since the clean_text function was not making the text lowercase, when the count_keywords function was called on the word_list, the accumulator would not include those few lexicon keywords, since those words wouldn't be found in WORD_TO_INTENSITY since all the words in WORD_TO_INTENSITY are all lowercase. So, the average intensity was calculated without all of the lexicon words, therefore an inaccurate average intensity was calculated.

   test_transformers failed because the count_keywords function was not properly accumulating the occurrences of lexicon keywords in the text. Prior to fixing the error, count_keywords would check each word in the word_list to see if it's in WORD_TO_INTENSITY, and if it is, it will see if it's not already in the dictionary accumulator. And if it isn't, the word will get added, and its value would be set to 0 and then get 1 added to it in a variable reassignment in the next line. This means that if there are duplicate words that are in WORD_TO_INTENSITY, nothing will happen to its value in the accumulator dictionary because there was no else statement for it the word is already in the dictionary. So, I changed the inner if-statement to be such that if the word is a WORD_TO_INTENSITY and it isn't in the dictionary, then it will be added and its' value will be set to 1, and if it the word is already in the dictionary, then its' value will be reassigned to what it was before, plus 1. Since the transformers review had the word 'terrible' twice, only the first terrible got accounted for and so the average intensity did not calculate the accurate value.

3. test_star_wars passed on the original code since the star wars review text did not have any lexicon words that had a capital letter in it and since each lexicon word only appeared once. Since all the lexicon words in the text were lowercase to begin with, the count_keywords function was able to find all the lexicon words in the text. And, since it didn't have any repeat words, the count_keywords properly accumulated the occurrences of lexicon words.

Part 3: Chaos, Fractals, Point Sequences

Complete this part in the provided `a3_part3.py` starter file. Do **not** include your solutions in this file.