**Anita Pal**
**Faculty of Arts and Humanities**
**Linnaeus University**
**ap224pu@student.lnu.se**

## Introduction

The following Python implementation, hosted on GitHub and Google Colab, contains functions for reading text files, removing stop words, counting the top 10 frequencies in those files, and creating visualisations using Pandas[1] and Seaborn[2].

## Problem Definition

Building on the previous assignment, this one engages Python *with* to open files, ensuring that resources are closed when no longer in use, even in the event of an error (Dataquest, 2025). However, as this assignment contained more than one instance that required a different UTF encoding (i.e., line 107), this assignment utilises a helper function for such a scenario (i.e. lines 40 to 47), making the code more readable and understandable while keeping other tasks from being cluttered with too many details (Matthes, 2023).

Additionally, this assignment uses several external Python modules that contain collections of code to solve problems in specific domains and often provide additional features. The advantages of using external libraries go beyond efficiency; they also include greater exposure for developers to an active community that can provide support (Raza). In Google Colab, the *!pip* command (i.e., line 12) installs external libraries (Stack Overflow user Ashutosh Pathak, 2018).

Pandas are usually employed in data analysis (Raza); however, for this assignment, they are utilised to create data frames (i.e. lines 122 to 123) -- structured formats that allow rows and columns to be constructed, similar to a spreadsheet (Miami University Center for Analytics and Data Science), not only enabling data manipulation, but also making it possible for data scientists to create visualisations (Day, 2025). Meanwhile, the csv module implements methods for reading and writing CSV files — the most common import and export format for spreadsheets and databases — in tabular format, as demonstrated in lines 97-101 of the s*ave_values_to_csv_file* function (Python Software Foundation, 2024).

## Methodology Approach

Some of the functions in this assignment (e.g., *create_stop_word_list_from_file* in lines 32 to 36) were already discussed in the report for the previous assignment, with one noteworthy addition: the helper function *open_file_and_handle_encoding* in lines 40 to 47, which handles files with different encodings. The function takes in the file path, a boolean value called *encoding*, and the encoding used itself as arguments. When the boolean *encoding* is set to true (i.e., line 41), the *encoding_used* variable is supplied as an argument to the *encoding* keyword on line 42, overriding the default Python encoding (Hunner, 2024), which allows the file contents to be read on line 43.

Otherwise, if no encoding is present, an else statement (i.e. line 44) opens the file in read mode without specifying an encoding (i.e. line 45), allowing the file contents to be read and returned on lines 46 and 47,

[1] https://pandas.pydata.org/docs/
[2] https://seaborn.pydata.orgfunction (i.e., lines 51 to 54), which may encounter files

respectively. While the function reuses some code (cf. Virtualik), it avoids duplication in areas such as the *convert_words_to_lowercase* function (i.e., lines 51 to 54), which may encounter files with an encoding that does not necessarily correspond to the expected standard. It is also used in the *create_stop_word_list_from_file* function, which uses a specific encoding for the stop word list (i.e., line 33).

The *process_text_for_frequency* function (lines 58-64) mentioned in the previous assignment's report cleans the text. This step often plays a vital role in data analysis by removing duplicates, errors, and other faults before additional processing (GeeksforGeeks, 2025a). In lines 89 to 93, the *get_top_ten_frequencies* function takes a cleaned word list and a book chapter as arguments, then calls the *get_all_frequencies* function, which takes the cleaned list and returns all the frequencies within it. Afterwards, as stated in the previous assignment, the output of that function is used to obtain the top ten frequencies (i.e., line 92), but not before making a call to the *store_frequency_based_data_dict* function on lines 77 to 85 using the top ten frequencies as an argument.

In line 78 in that function (GeeksforGeeks, 2025b), an empty list called *rows* is created, which is then used in a for loop that iterates through the top ten words and frequencies one by one (i.e. line 79) before appending these values into the previously empty row list to store the data in a specific format, while also retrieving the length of a given word (i.e. line 83) and chapter of the book (i.e. line 84).

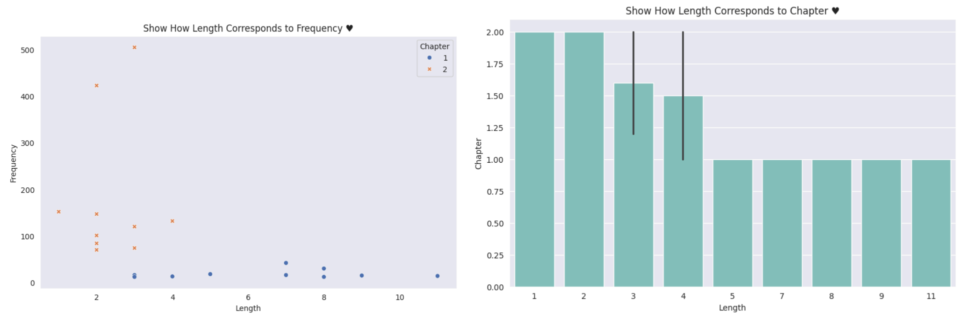
The *save_values_to_csv_file* function takes a file output and the rows mentioned in the previous paragraph as inputs, then opens a CSV file on line 98. In the CSV file, line 99 creates a variable called *writer* that writes the rows to the CSV file, along with the columns (which are declared as a separate variable on line 28). The columns are written first on line 100, before all the rows on line 101 (Fincher).

In lines 104 to 117, which deal with a) the reading of the stop word list into a list (i.e. line 104), b) cleaning the first chapter by removing its stopwords and converting its content to lowercase (i.e. line 107) and then saving the results of the top five frequencies into a CSV file (i.e. line 109) c) doing the same for second chapter in lines 112 to 114 result in a CSV file that contains all the necessary data to create specific visualisations (i.e. line 118).

In line 122, a data frame is created from the CSV file on line 118 (GeeksforGeeks, 2025c). In line 123, the head method returns the first five rows of the data frame (pandas development team, 2024). Then, the first visualisation is created on line 134 using Seaborn's *scatterplot* method (GeeksforGeeks, 2025d), with the previous lines giving the graph a unique look and feel, such as custom colours (i.e., line 129) (Seaborn development team, 2024).

Similarly, in line 144, Seaborn's *barplot* method is used to create the second visualisation (GeeksforGeeks, 2025d), which uses customisations of its own, such as a unique style via the *set_style* method (Seaborn development team, 2024).

## Analysis of Results



Figures 1 and 2 show a scatter plot and a bar plot, respectively. *Note*. Screenshots.

Figure 1 shows how the various lengths map to the frequencies for both chapters. In Figure 2, a box chart shows that the lengths of the boxes correspond to the chapters in both books in equal proportions. For both graphs, the focus was aesthetics rather than mathematical precision.

## Conclusions and Reflections

A fun assignment that allowed me to dabble in visualisations again.

(1021 words, including in-text references but excluding titles).

## AI Acknowledgements

I only use AI to generate APA citations and to ensure no embarrassing typos make it into my report.
The code is mine and mine alone, with credit being given where it is due (and also visible in previous assignments for this module).

## References

Dataquest. (2025, April 7). *Tutorial: How to easily read files in Python (text, CSV, JSON).* https://www.dataquest.io/blog/read-file-python/

Day, F. (2025, July 15). *Why every data scientist should know Pandas DataFrames.* Noble Desktop. https://www.nobledesktop.com/blog/why-learn-pandas-dataframes-for-data-science

Fincher, J. (n.d.). *Reading and writing CSV files in Python.* Real Python. https://realpython.com/python-csv/

GeeksforGeeks. (2025a, September 16). *Data cleaning in ML.* https://www.geeksforgeeks.org/data-analysis/data-cleansing-introduction/

GeeksforGeeks. (2025b, July 11). *Ways to create a dictionary of lists – Python.* https://www.geeksforgeeks.org/python/python-ways-to-create-a-dictionary-of-lists/

GeeksforGeeks. (2025c, July 11). *Pandas read CSV in Python.* https://www.geeksforgeeks.org/pandas/python-read-csv-using-pandas-read_csv/

GeeksforGeeks. (2025d, December 10). *Data visualization with Seaborn – Python.* https://www.geeksforgeeks.org/data-visualization/data-visualization-with-python-seaborn/

Hunner, T. (2022, May 2). *Unicode character encodings.* Python Morsels. https://www.pythonmorsels.com/unicode-character-encodings-in-python/

Matthes, E. (2023, August 31). *OOP in Python, part 9: Helper methods.* Mostly Python. https://www.mostlypython.com/oop-in-python-part-9-helper-methods/

Miami University Center for Analytics and Data Science. (n.d.). *Pandas dataframes.* https://miamioh.edu/centers-institutes/center-for-analytics-data-science/students/coding-tutorials/python/pandas-dataframes.html

pandas development team. (2024). *pandas.DataFrame.head..* In *pandas 2.3.3 documentation.* https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.head.html

Python Software Foundation. (2024). *csv — CSV file reading and writing.* In *Python documentation* (Version 3.14.2). https://docs.python.org/3/library/csv.html

Raza, D. (n.d.). *Using external libraries in Python.* Medium. https://medium.com/@dealiraza/using-external-libraries-in-python-dba5087de047

Seaborn development team. (2024). *seaborn.scatterplot..* In *Seaborn 0.13.2 documentation.* https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Stack Overflow user Ashutosh Pathak. (2018, July 14). *How do I install Python packages in Google's Colab?* [Answer to the question "How do I install Python packages in Google's Colab?"]. Stack Overflow. https://stackoverflow.com/a/51342586

Virtualik. (n.d.). *DRY (Don't Repeat Yourself) principle in Python: A practical guide.* Medium. https://medium.com/@virtualik/dry-dont-repeat-yourself-principle-in-python-a-practical-guide-06290ebda0cf