**COMP130 Lab 10: Final project**

**Stylometric Authorship Attribution of the Federalist Papers  
instructor: John MacCormick**

*This lab is worth 160 points. It is intended to serve as a final project for the course. You may choose to complete this lab on your own or with a partner of your choice. The last week of classes and the final week’s lab session will be available for work on this lab.*

*The ideas for this project are adapted primarily from Michelle Craig’s Authorship Detection assignment. Detailed acknowledgments are provided at the end of this document.*

The Federalist Papers is a collection of 85 essays published in the 1780’s, arguing in support of the newly proposed constitution for the United States of America. They were published under the pseudonym Publius, but it’s known that the essays were written collectively by Alexander Hamilton, James Madison, and John Jay. For most of the essays, the authorship is known. However, the exact authorship of twelve of the papers has been widely disputed. In this project you will write code that uses some elementary stylometric analysis techniques to attempt to attribute the authorship of the disputed Federalist Papers to Hamilton, Madison or Jay.

The texts of the Federalist Papers are contained in the papers folder provided with the project materials. Copy this folder to your lab10 folder. Examine the files there.

Qu 1. [Answer in responses document] (5 points) How many files are in the papers folder? What is the naming convention of the files? What is the file name of Federalist No. 26?

Create a new Python file called authorship.py in your lab10 folder. This file will be submitted together with your responses document at the end of the lab.

Qu 2. (5 points) Write a function get\_paper\_name(n). The parameter n is an integer specifying the number of a Federalist paper, between 1 and 85 inclusive. The function returns the filename of the corresponding Federalist paper. The filename must include the full path from your lab10 folder to the desired file. For example, the return value for paper 15 should be 'papers/federalist\_15.txt'. To test your function, copy the function test\_get\_paper\_name() from the authorship\_tests.txt file provided with this lab.

Qu 3. (5 points) Write a function get\_paper\_text(n). The parameter n is an integer specifying the number of a Federalist paper, between 1 and 85 inclusive. The function returns the complete text of the corresponding Federalist paper as a single string. Use the provided test\_get\_paper\_text() to check for correctness. For full credit, make sure to close any file that is opened by this function.

For our first simple attempt at analyzing the authorship of the Federalist papers, we will investigate the average word length used by each author. For example, we might hypothesize that Madison tends to use longer words than Hamilton or Jay. If so, the average word length of a paper may give us a clue as to its authorship.

Qu 4. (10 points) Write a function average\_word\_length(words). The parameter words is a list of strings. The function returns the average length of the strings in the list. Use the provided test\_average\_word\_length() to check for correctness.

Qu 5. (10 points) Write a function paper\_avg\_word\_len(n: int). The parameter n is an integer specifying the number of a Federalist paper, between 1 and 85 inclusive. The function returns the average length of the words in the corresponding Federalist paper. Use the string split() method to split the contents of a paper into words. This will result in words that include punctuation and numbers, such as 'society.' and 'circumstance,'. Hence, the computed average length of the words will not be exactly correct. However, we prefer to take a simple approximate approach here. Please do not try to improve the accuracy; just use the results of the split() method. Use the provided test\_paper\_avg\_word\_len() to check for correctness.

Qu 6. (10 points) Write a function collection\_avg\_word\_len(paper\_nums). The parameter paper\_nums is a list of integers specifying a collection of Federalist papers. For example, [1,3,5] refers to a collection of Federalist papers 1, 3, and 5. The function returns the average length of the words in the collection. More accurately, the return value is an average of the average word lengths in each paper of the collection. (Yes, this is an average of averages. First, we obtain the average word length of each paper in the collection. Then we take the average of those numbers.) Use the provided test\_collection\_avg\_word\_len() to check for correctness.

Here are some Python variables defining collections for the papers whose authorship is known and unknown:

jay\_papers = [2, 3, 4, 5, 64]

madison\_papers = [10, 14, 18, 19, 20, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48]

hamilton\_papers = [1, 6, 7, 8, 9, 11, 12, 13, 15, 16, 17, 21, 22, 23, 24,

25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 59, 60,

61, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77,

78, 79, 80, 81, 82, 83, 84, 85]

unknown\_papers = [49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 62, 63]

Qu 7. [Answer in responses document] (30 points) Use your existing Python code, together with any new code that may be necessary, to compute the average word length for the collections of papers by the three known authors. State your results clearly, ordering the authors from smallest to largest average word length. (Do not submit any Python code for this question.)

If your answers to the previous question were correct, you found that Hamilton has the shortest words, Jay is in the middle, and Madison has the longest words on average. Let’s call these average numbers Now suppose we are given a new Federalist paper that has an average word length of 4.1 characters. What’s our best guess of the author, based only on this information? Well, 4.1 is smaller than Hamilton’s average , which is smaller than the others. So we can’t be sure, but Hamilton is probably the best guess. Now suppose we are given a new Federalist paper with an average word length , somewhere between and . If we are forced to guess the author based only on this information, it seems reasonable to guess whichever value is closer. That is, if is closer to than , guess Hamilton and otherwise guess Jay. Mathematically, we guess Hamilton if . Similar reasoning applies to values between and .

Qu 8. (10 points) Write a function guess\_paper\_by\_word\_len(n), where n is the number of a Federalist paper. The function should use the method suggested above to guess the author of the corresponding Federalist paper. It should return one of the three strings 'Hamilton', 'Madison', or 'Jay'. Use the provided test\_guess\_paper\_by\_word\_len() to check for correctness.

Qu 9. [Answer in responses document] (15 points) Use your existing Python code, and any new code as necessary, to conduct the following experiments. For each of the five papers that is known to be by Jay, guess the author using the function from the previous question. State your results clearly. Provide one or two sentences of additional commentary, noting how many of these guesses were correct. State how many guesses would have been correct if you randomly guessed one of the three authors each time. Comment on whether our method for guessing the paper author based on average word length appears to be better than random guessing.

Qu 10. [Answer in responses document] (15 points) Repeat the previous question, but this time conduct your experiments on the 17 paper is known to have been authored by Madison. State your results clearly and provide the same type of commentary as in the previous question.

Qu 11. [Answer in responses document] (5 points) Write one or two sentences explaining whether or not you would recommend determining the authorship of these papers using average word length.

Qu 12. (10 points) Write a function get\_word\_counts(words), where words is a list of strings. The function returns a dictionary whose keys are strings and values are integers. The dictionary represents the number of times each word appeared in the list. For example, if the list of words is ['a', 'b', 'a'] then the returned dictionary has two key-value pairs: 'a' maps to 2, and 'b' maps to 1. Use the provided test\_get\_word\_counts() to check for correctness.

In text analysis, the *type-token ratio* is the ratio of the number of distinct words in a text to the total number of words in that text. Example: For the phrase “the orange cat sat on the orange mat,” there are 6 distinct words and 8 words in total, so the type-token ratio is 6/8 = 0.75. We might hypothesize that the type-token token ratio can be used to guess the authorship of a text.

Qu 13. (10 points) Write a function type\_token\_ratio(words), where words is a list of strings. The function returns the type-token ratio of the list. Example: For the input ['the', 'orange', 'cat', 'sat', 'on', 'the', 'orange', 'mat'] the return value is 0.75. Use the provided test\_type\_token\_ratio() to check for correctness. Hints: 1. Use the function from the previous question as a subroutine. 2. The number of keys in a Python dictionary d is len(d).

Qu 14. [Answer in responses document] (15 points) Use your existing Python code, together with any new code that may be necessary, to compute the average type-token ratio for the collections of papers by the three known authors. State your results clearly, ordering the authors from smallest to largest average type-token ratio. (Do not submit any Python code for this question.)

Qu 15. [Answer in responses document] (5 points) In two or three sentences, state whether the type-token ratio is likely to be better or worse than the average word length for guessing the authorship of a paper, and explain your reasoning. You are not required to perform any experiments. Just give your hypothesis and reasoning.

# Optional questions

*The remaining portions of this lab are worth a small amount of extra credit for each question. Please consider the rest of the lab optional, and only pursue these questions if you have ample time to complete your other work for this course and other courses.*

Qu 16. [Answer in responses document] Conduct experiments to investigate whether the type-token ratio is superior to the average word length for guessing the authorship of the Federalist papers. State your results and conclusions.

Qu 17. [Answer in responses document] The *hapax legomena ratio* is the ratio of words that occur exactly once in the text to the total number of words in the text. Example: For the phrase “the orange cat sat on the orange mat,” there are 4 words that occur exactly once and 8 words in total, so the type-token ratio is 4/8 = 0.5. Write code that compute the average hapax legomena ratio for each author, and investigate whether this metric is useful for guessing the authorship.

Qu 18. [Answer in responses document] Write code that combines all three metrics used in this lab (average word length, average type-token ratio, average hapax legomena ratio) into a single score for identifying authorship. Specifically, suppose the three metrics for a given author are respectively for word length, type-token ratio, and hapax legomena ratio. Suppose we are given a new paper , and the three metrics for that paper are Define the *total relative error* to be the sum of the percentage errors for each of these metrics:

Write code that computes the total relative error for each of the three authors, when given a particular paper as input. Use this to investigate whether the total relative error is more useful for guessing the authorship of the Federalist papers, compared to the single metrics investigated so far.

Qu 19. [Answer in responses document] Based on all of your results, write code to guess the authorship of the 12 papers whose authorship is unknown. As stated earlier,

unknown\_papers = [49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 62, 63]

State your results clearly. Do some brief online research to determine the authorship of the unknown papers that is generally agreed by present-day scholars. Compare these conclusions to your own results.

Qu 20. [Answer in responses document] Improve your results by using any of the following techniques:

* Eliminate numbers and punctuation from all words.
* Convert all words into lowercase before analyzing.
* Use import nltk to import the Natural Language Processing Tool Kit, then use nltk.tokenize.word\_tokenize() to extract the words from each paper, instead of the str.split() method we used earlier.
* Incorporate *sentence length* as a new metric for stylistic analysis. Use nltk.tokenize.sent\_tokenize() to break each paper into sentences. Then use word\_tokenize() to break each sentence into words, and thus compute the average number of words per sentence for each author.

Describe your results in the responses document.

# Additional Information

* The authorship of the disputed papers remained unsettled until the mid 1900's. Since then, a combination of historical detective work and computer analysis has largely settled the authorship. [Douglass Adair](https://en.wikipedia.org/wiki/Douglass_Adair) in his article [The Authorship of the Disputed Federalist Papers](https://www.jstor.org/stable/1921883?seq=1) attributed all of the disputed works to Madison. This attribution was reinforced by Mosteller and Wallace who used a computer to do stylometric analysis as described in their 1964 book *Inference and Disputed Authorship: The Federalist*.
* Despite the above work, the site www.congress.gov, "the official website for U.S. federal legislative information" which "provides access to accurate, timely, and complete legislative information for Members of Congress, legislative agencies, and the public" still lists some of the [Federalist Papers](https://www.congress.gov/resources/display/content/The+Federalist+Papers) with shared and/or disputed authorship.
* Stylometric approaches have been used in the real-world to
  + make a convincing case that an [unattributed play was co-written by Shakespeare](http://content.time.com/time/arts/article/0,8599,1930971,00.html?artId=1930971?contType=article?chn=arts).
  + do [fraud and plagarism detection](https://pdfs.semanticscholar.org/0df1/d4097c970483dac9b09bd65df1a2d397074b.pdf) for on-line learning systems.
  + filter out and identify spearfishing e-mails.
  + identify the [authors of tweets](https://link.springer.com/chapter/10.1007/978-3-319-03689-2_3).
  + [link multiple social media accounts owned by the same person](https://www.osti.gov/servlets/purl/1456316).
  + provide evidence in [an immigration case](https://brooklynworks.brooklaw.edu/cgi/viewcontent.cgi?article=1043&context=jlp)

# Acknowledgments

* This lab is adapted primarily from Michelle Craig at the University of Toronto and her assignment [Authorship Detection](http://nifty.stanford.edu/2013/craig-authorship-detection/handout/index.shtml) as contained in the [Nifty Assignments](http://nifty.stanford.edu/) repository.
* Parts of the lab are taken from another Dickinson College version by Grant Braught.
* Using the Federalist papers as a compelling dataset was inspired by François Dominic Laramée's ["Introduction to stylometry with Python"](https://programminghistorian.org/en/lessons/introduction-to-stylometry-with-python) in [The Programming Historian](https://programminghistorian.org/) #7 (2018).
* The text of the Federalist Papers was obtained from Laramée's version, which was in turn obtained by splitting the complete collection available on [Project Gutenberg](http://www.gutenberg.org/ebooks/18) into the individual documents.