Text Analysis: What Makes a Great Textbook

Arturo Munoz, Harry Hillsdownley, Chan Kang, Christian Kosinski,
David Lee, Manuel Rocha

May 6th, 2025

TABLE OF CONTENTS

1 Introduction

Textbooks are great resources that provide deep insight into a wide variety of academic areas. Particularly, in college, textbooks are often used to structure the material and outline of courses, and students can use them as a tool to supplement their professor's lectures, to prepare for exams, or even to dive deeper into particular areas of interest. Consequently, the quality of a textbook plays a crucial role in a student's education. This leads to some crucial questions. What makes one textbook better than another? What are some qualities of highly regarded textbooks? Does the textbook's academic subject matter influence which characteristics are important?

In this paper, we sought to answer these questions by examining a large collection of textbooks using text mining and Natural Language Processing with the help of the High Performance Computing Cluster. We wanted to be able to extract information and characteristics directly from the textbooks themselves, compare these characteristics, and see how they may influence the ratings of the textbooks.

For the textbooks, we accessed the "Open Textbook Library" from the University of Minnesota that featured over 1500 textbooks that are openly licensed. From this database, the textbooks also received ratings from faculty. These ratings are what we used to determine the textbooks' consensus qualities. As a result, our findings may help us determine what makes a textbook good in the eye's of professors, but this may not directly translate to students' perspectives.

One feature we wanted to determine using Natural Language Processing was the textbook's readability. Would a textbook that is easier to read be better since it would be easier to understand, or would its simplicity result in a textbook that is not rich enough with information and detail? A harder to read textbook may appear to be better due to it having more challenging and academic prose, but could this increase in complexity reduce the overall educational utility of the textbook? To determine readability, we used the Flesch Reading Ease Test that was made by Rudolf Flesch. This produces a score using a formula that takes the total number of words, syllables, and sentences into consideration [1]. The formula is given as:

$$Flesch\ Reading\ Ease = 206.835 - 1.015\left(\frac{total\ words}{total\ sentences}\right) - 84.6\left(\frac{total\ syllables}{total\ words}\right)$$

With this readability score, we will try to identify any patterns with how it relates to the textbooks' ratings and other features.

Two other features we wanted to determine were the textbooks' lexical diversity and average word length. The lexical diversity was determined by dividing the number of unique words by the total number of words. With these features, we aim to find a correlation that helps explain what characteristics result in better textbooks.

2 Tools Used

2.1 Python

The project leveraged Python 3.12 for all scripting tasks, including data collection, preprocessing, and analysis. Key Python libraries utilized in the pipeline included:

- Requests and BeautifulSoup for web scraping and API interactions.
- PyPDF for extracting text from PDF documents.
- Concurrent.futures for parallel processing during data collection and text extraction.
- JSON for metadata serialization and ledger management.
- SpaCy (see Section 2.3) for natural language processing (NLP) tasks.
- Pandas for the reading of .csv files and data frame manipulations

- Seaborn and Pyplot from Matplotlib to produce graphs for the data visualization

Python's versatility enabled efficient integration with high-performance computing (HPC) workflows, particularly through its support for multiprocessing and cluster job scheduling.

2.2 Secure Copy

Data transfer between local systems and the Markov HPC cluster (markov.case.edu) was managed via Secure Copy Protocol (SCP), a cryptographic network protocol for secure file transfers. The workflow included:

- Bulk transfers of raw PDFs and processed text files to cluster storage.
- Script synchronization to ensure consistent execution environments.
- Output retrieval for post-processing and visualization.

2.3 SpaCy

For NLP tasks, we employed SpaCy's pretrained transformer model (en_core_web_trf) to:

- Tokenize and annotate textbook content.
- Extract semantic features (e.g., named entities, syntactic dependencies).
- Enable downstream tasks like topic modeling and keyword analysis.

The model's GPU-optimized architecture aligned with our HPC requirements, allowing efficient batch processing of large text corpora.

2.4 R

For data modification and visualization purposes, R was utilized. The key packages utilized throughout the process were:

- Tidyverse and ggplots are used for vivid visualization of bar graphs and scatter plots with best-fit lines.
- Hrbrthemes for graph theme customization

R provides a powerful visualization tool like ggplot2, allowing us to generate high-quality or interactive plots. These functions can also be operated as part of batch jobs or remote sessions. These capabilities make R well-suited for large-scale data analysis in HPC settings.

3 Data Collection

The dataset for this project was sourced from the University of Minnesota's Open Textbook Library (open.umn.edu), a comprehensive repository hosting over 1,600 openly licensed textbooks across a wide range of academic disciplines. To support large-scale text analysis and machine learning tasks, we designed a robust, automated data collection pipeline that extracted both metadata and content from this resource with minimal manual intervention.

Our process began with identifying a suitable method for programmatic access. Fortunately, the Open Textbook Library featured a search function that allowed us to issue a blank query—returning the entire catalog in a paginated format. By analyzing the network activity of the client-side application, we discovered that textbook entries were retrieved via a paginated GET request to an XML feed endpoint. Further inspection revealed that two query parameters—search and page—were responsible for filtering and navigating the results. By passing a blank search string and incrementally increasing the page index, we were able to automate complete traversal of the collection using a single endpoint.

A custom crawler was developed to paginate through the API, constructing requests and retrieving XML data until no further entries were returned. To make this data more manageable, we converted the XML response into JSON-like Python dictionaries using the xmltodict library, resulting in a structured list of dictionaries where each dictionary represented a textbook record. These entries included core fields such as title, author, publication year, and links to the textbook's web page and available formats.

However, the API metadata was incomplete in several key areas. To enrich our dataset, we implemented an additional scraping step using BeautifulSoup4. For each textbook page, our scraper extracted standardized HTML elements containing extended metadata—such as user ratings, format availability, language, publisher, and ISBN. These fields were parsed by identifying specific HTML classes and extracting their contents using BeautifulSoup's tag traversal utilities. This two-stage collection strategy ensured we had a comprehensive and consistent dataset.

An important step in our pipeline was securing access to the full content of the textbooks. Although some metadata entries linked directly to PDF files, many instead pointed to intermediary hosting pages. To resolve this, we developed a lightweight URL-crawling algorithm that attempted to locate actual download links. For each textbook entry, the algorithm ranked all outbound URLs by the likelihood of pointing to a downloadable PDF, checking only the top ten candidates per textbook to maintain performance. When automated extraction failed—approximately 200 cases in total—manual intervention was used to acquire the PDFs and include them in the dataset.

Given the scale of this operation and the server-side rate limitations of the Open Textbook Library, we implemented a rate-limiting strategy to prevent throttling or service disruptions. Our scraper was configured with a request rate of 0.9 requests per second (i.e., a delay of approximately 1.1 seconds between requests). If a response was detected to contain the text "Retry later"—a heuristic for server-imposed rate limiting—the script paused for four minutes before resuming. This defensive approach allowed us to complete data acquisition without violating the host's usage policies.

Once the collection phase was complete, we shifted to preprocessing the documents for analysis. Each textbook was parsed using pypdf with a layout-preserving parser to convert its contents into plain text. We applied quality control filters to discard files that failed to extract properly or had insufficient content—defined as fewer than 10,000 characters. Textbooks that passed the filter were then synchronized with their metadata through content checksums, ensuring data integrity across all stages of the pipeline. Failed downloads and parsing errors were logged for transparency and reproducibility.

We then filtered the extracted textbooks to make sure that they had at least 10,000 characters of content as well as at least one rating to ensure data quality.

The final dataset consisted of 1,329 high-quality textbooks with complete metadata and validated content, providing a rich foundation for our subsequent natural language processing and machine learning tasks.

4 Program

The program was implemented with the purpose of deriving insights from the data to perform data visualization for further processes. The insights implemented were the Flesch Reading Ease score, lexical diversity, and categorical hierarchies for the textbooks in our database. To implement these objectives, our program was split into the following sections:

1. Categorizer: Compute the categorical hierarchy of the textbooks

2. Sentence Tokenizer: tokenize the textbooks into sentences to rank their lexical complexity
3. Lexical Complexity Analyzer: Use Natural Language Processing (NLP) to compute readability scores, lexical diversity, and average word length for each of the tokenized sentences

For successfully implementing these programs, finding the right NLP model was important, as sentences can be difficult to rank depending on their grammatical dependencies and structure. This is why we decided to implement Natural Language Processing in our program for tokenizing sentences.

4.1 NLP Process

SpaCy is an open-source Python library designed for Natural Language Processing, that features tokenization, part-of-speech tagging, dependency parsing, and lemmatization. SpaCy is at the core of our NLP Processes, as sentences in academic textbooks involve complex grammatical structures that cannot be tokenized with a pattern, such as tokenizing at periods within sentences.

This is why this paper decided to use the en-core-web-sm model within SpaCy to intelligently perform sentence tokenization on the textbooks to facilitate our process of calculating metrics on each sentence for every textbook, which could be generalized to the lexical complexity and features of the whole corpus.

4.2 Implementation

For our implementation, this paper took advantage of the HPC to compute a large range of data, as we were running NLP on every sentence in 1,600 academic-level textbooks. This implementation featured 3 highly-performant, multithreaded scripts, which took advantage of all available cores to perform the computations for NLP metric generation.

The first script was the tokenizer, which used the chosen NLP implementation to break up textbooks into sentences. The second script was a multithreaded lexical complexity computation script that calculated the lexical complexity for each sentence in the textbook corpus. Finally, the third script performed title-to-structure keyword matching to categorize the textbooks into areas, subsections, and topics within academia, although a large number of textbooks were not able to be categorized.

Through this implementation, this paper was able to obtain a robust, modular, and highly performant structure to generate data to visualize dependencies between lexical complexity and ratings across a large corpus of academic textbooks.

5 Markov Cluster Computation

To process the large corpus of textbooks in a computationally efficient manner, we leveraged the High Performance Computing (HPC) resources available through the Markov cluster at Case Western Reserve University. Specifically, we utilized a compute node with 32 cores and 40GB of RAM to parallelize and expedite the computation of our textual metrics across all 1,329 textbooks.

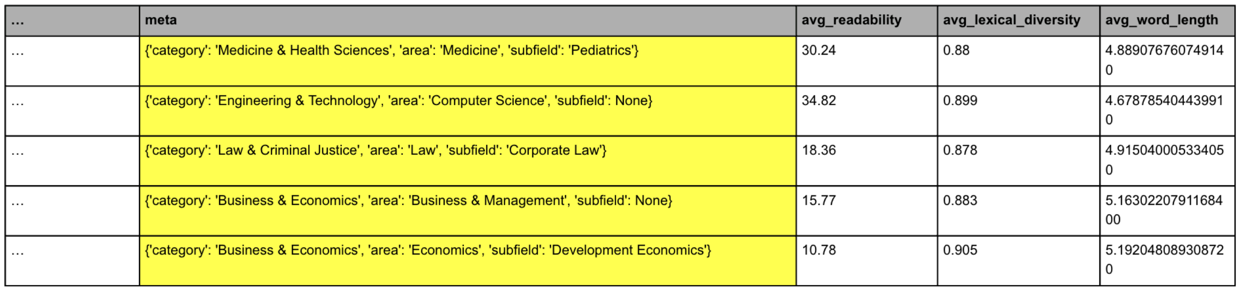
The Markov cluster enabled us to efficiently run resource-intensive tasks such as text parsing, readability computation, lexical diversity analysis, and average word length extraction. These operations, which involved parsing millions of words and computing sentence-level statistics across a diverse range of documents, were implemented in Python using multiprocessing and job scheduling frameworks. The parallel execution across 32 cores dramatically accelerated our workflow, allowing us to complete all metric computations in just under 50 minutes.

For comparison, preliminary benchmarks conducted on a standard consumer-grade laptop (with a quad-core processor and 8GB of RAM) estimated that the same process would have taken approximately 20 hours to complete sequentially. This highlights the transformative impact of HPC resources in scaling up data analysis pipelines and managing large academic datasets effectively.

By batching the data and distributing workload across multiple processes, we were able to reduce I/O bottlenecks and optimize memory usage—critical factors when working with large volumes of plain text and intermediary outputs. The use of the Markov cluster not only improved our processing speed but also ensured the reproducibility of our results through controlled computing environments and isolated execution contexts.

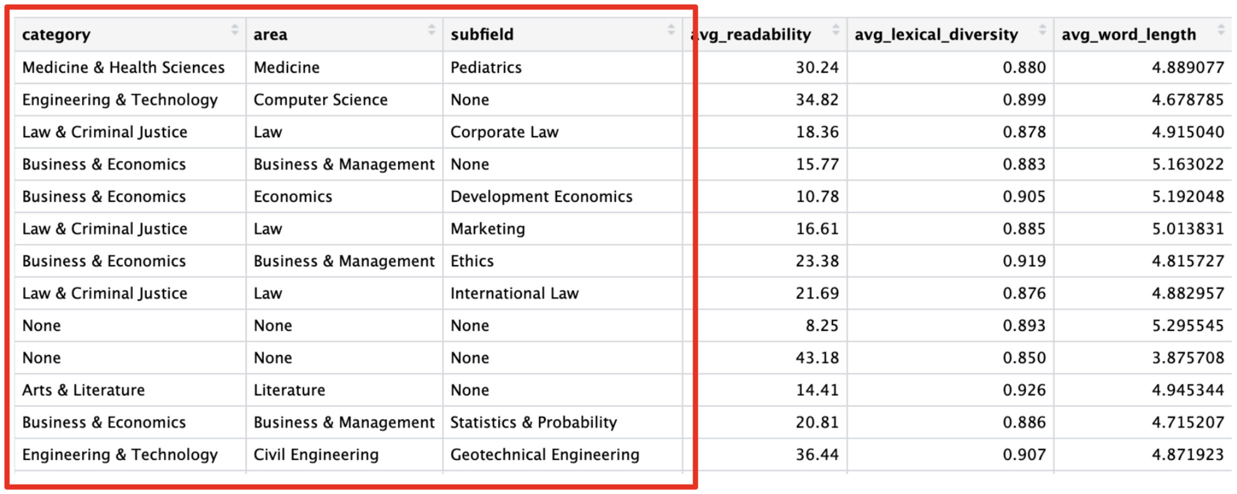
In addition to computational efficiency, the Markov cluster facilitated streamlined data management throughout our project. Leveraging its robust infrastructure, we efficiently handled the transfer of large volumes of textbook data and associated metadata. This capability ensured that our preprocessing and analytical workflows operated seamlessly, minimizing potential delays and bottlenecks. The cluster's support for secure and efficient data handling was instrumental in maintaining the integrity and consistency of our dataset, thereby enhancing the reliability of our subsequent analyses.

6 Data Modification



...	meta	avg_readability	avg_lexical_diversity	avg_word_length
...	{'category': 'Medicine & Health Sciences', 'area': 'Medicine', 'subfield': 'Pediatrics'}	30.24	0.88	4.889076760749140
...	{'category': 'Engineering & Technology', 'area': 'Computer Science', 'subfield': None}	34.82	0.899	4.678785404439910
...	{'category': 'Law & Criminal Justice', 'area': 'Law', 'subfield': 'Corporate Law'}	18.36	0.878	4.915040005334050
...	{'category': 'Business & Economics', 'area': 'Business & Management', 'subfield': None}	15.77	0.883	5.1630220791168400
...	{'category': 'Business & Economics', 'area': 'Economics', 'subfield': 'Development Economics'}	10.78	0.905	5.192048089308720

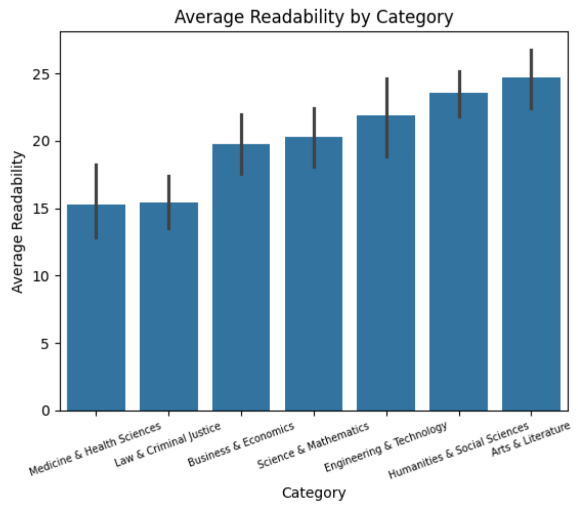
Using Python, we combined the textbook information and individual complexity measurements into one CSV file. Then, we found that the ‘meta’ column, indicating the field of textbooks, has three different entries: category, area, and subfield. Using R, we divided ‘meta’ into three different columns shown below.

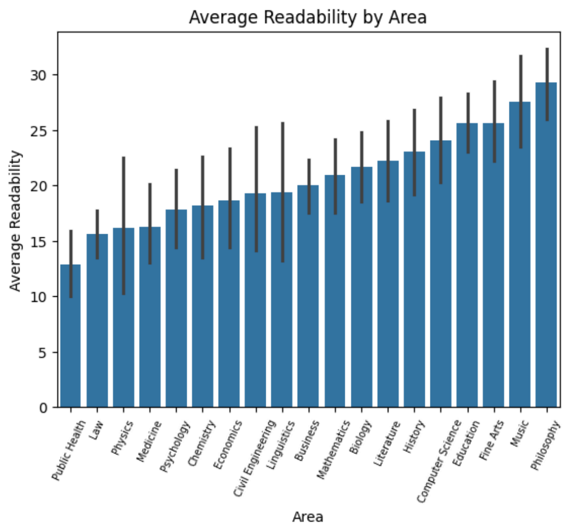


category	area	subfield	avg_readability	avg_lexical_diversity	avg_word_length
Medicine & Health Sciences	Medicine	Pediatrics	30.24	0.880	4.889077
Engineering & Technology	Computer Science	None	34.82	0.899	4.678785
Law & Criminal Justice	Law	Corporate Law	18.36	0.878	4.915040
Business & Economics	Business & Management	None	15.77	0.883	5.163022
Business & Economics	Economics	Development Economics	10.78	0.905	5.192048
Law & Criminal Justice	Law	Marketing	16.61	0.885	5.013831
Business & Economics	Business & Management	Ethics	23.38	0.919	4.815727
Law & Criminal Justice	Law	International Law	21.69	0.876	4.882957
None	None	None	8.25	0.893	5.295545
None	None	None	43.18	0.850	3.875708
Arts & Literature	Literature	None	14.41	0.926	4.945344
Business & Economics	Business & Management	Statistics & Probability	20.81	0.886	4.715207
Engineering & Technology	Civil Engineering	Geotechnical Engineering	36.44	0.907	4.871923

7 Data Visualization

Using R and Python, we were able to create graphs using the tabular data that we calculated and collected. Before creating any graphs for analysis, entries with outliers were removed. We defined an entry as an outlier if it had a feature that was greater than three standard deviations away from the feature's mean. We first compared the textbooks' categories and areas with its readability.

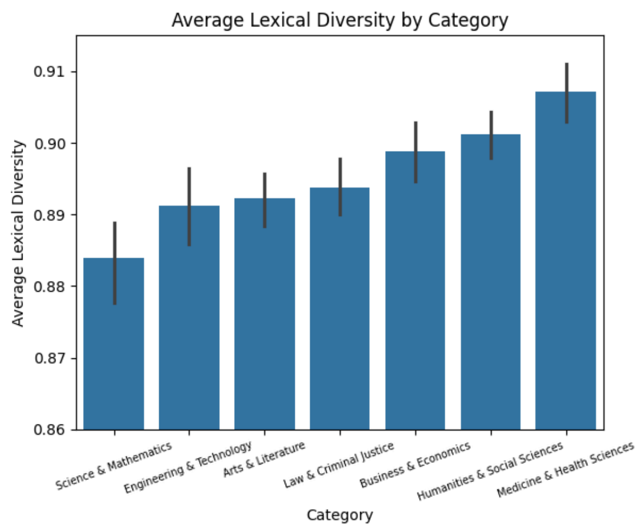


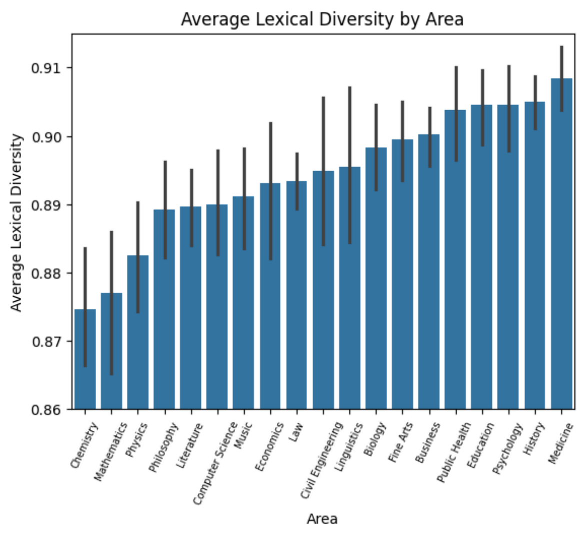


From these graphs, we noticed that a textbook's category can be indicative of the textbook's readability score. Using R to create a linear module across these two features, it was found that both Medicine & Health Sciences and Law & Criminal Justice had significantly different readability scores compared to the other textbooks. Those two categories had p-values of 1.42e-06 and 7.79e-06, respectively.

```
Call:
lm(formula = avg_readability ~ factor(category), data = df_cleaner)

Residuals:
    Min      1Q  Median      3Q     Max 
-22.344  -7.899  -0.754   6.824  63.156 

Coefficients:
                                             Estimate Std. Error t value Pr(>|t|)    
(Intercept)                                   22.8044     0.5449  41.849  < 2e-16 ***
factor(category)Arts & Literature              1.6497     1.1595   1.423   0.1551    
factor(category)Business & Economics          -3.0626     1.2944  -2.366   0.0181 *  
factor(category)Engineering & Technology      -0.9419     1.6364  -0.576   0.5650    
factor(category)Humanities & Social Sciences   0.5219     1.0466   0.499   0.6181    
factor(category)Law & Criminal Justice        -7.3718     1.5204  -4.849 1.42e-06 ***
factor(category)Medicine & Health Sciences    -7.5377     1.6779  -4.492 7.79e-06 ***
factor(category)Science & Mathematics         -2.3499     1.2379  -1.898   0.0579 .  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.44 on 1095 degrees of freedom
Multiple R-squared:  0.04752,	Adjusted R-squared:  0.04143 
F-statistic: 7.804 on 7 and 1095 DF,  p-value: 3.05e-09
```

Next, we compared the textbooks' categories and areas with their lexical diversity in a similar fashion.

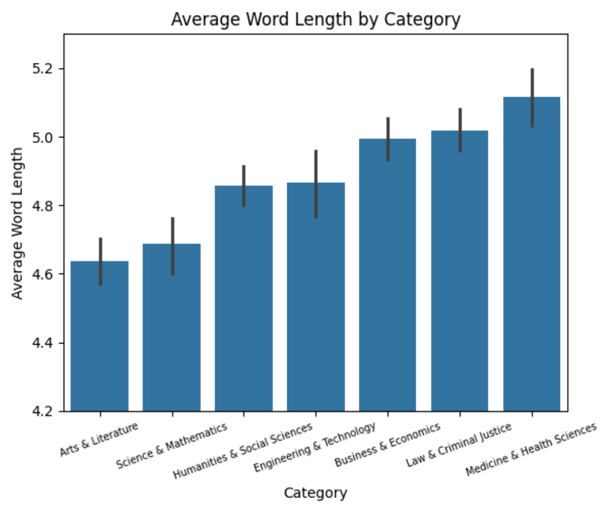


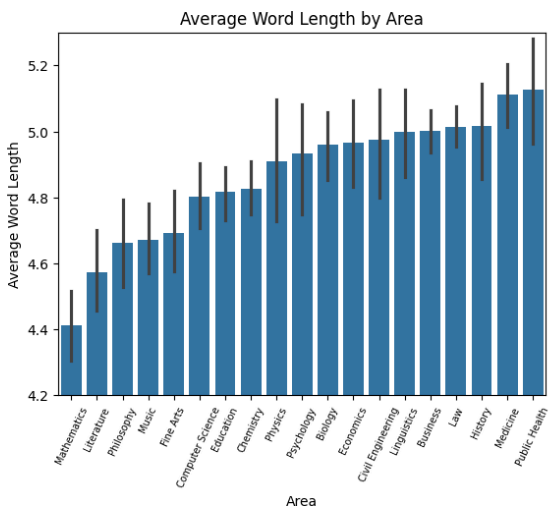


Again, we notice that the subject matter of the textbook has an impact on its complexity. This time, with Medicine & Health Sciences having a larger lexical diversity to indicate a more difficult text. Additionally, Science & Mathematics was found to have a significantly smaller lexical diversity than other texts, with a p-value of 1.75e-13.

```
Call:
lm(formula = avg_lexical_diversity ~ factor(category), data = df_cleaner)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.083524 -0.010705  0.001476  0.013231  0.058295 

Coefficients:
                                              Estimate Std. Error t value Pr(>|t|)    
(Intercept)                                  0.9027052  0.0009434 956.903  < 2e-16 ***
factor(category)Arts & Literature           -0.0096252  0.0020074  -4.795 1.85e-06 ***
factor(category)Business & Economics        -0.0038526  0.0022408  -1.719 0.085841 .  
factor(category)Engineering & Technology    -0.0115234  0.0028329  -4.068 5.09e-05 ***
factor(category)Humanities & Social Sciences -0.0011808  0.0018119  -0.652 0.514727    
factor(category)Law & Criminal Justice      -0.0089360  0.0026321  -3.395 0.000711 ***
factor(category)Medicine & Health Sciences   0.0043717  0.0029047   1.505 0.132599    
factor(category)Science & Mathematics       -0.0159882  0.0021430  -7.461 1.75e-13 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.01981 on 1095 degrees of freedom
Multiple R-squared:  0.07708,	Adjusted R-squared:  0.07118 
F-statistic: 13.06 on 7 and 1095 DF,  p-value: 3.135e-16
```

Lastly, we again compared the categories and areas to the textbooks' word length,

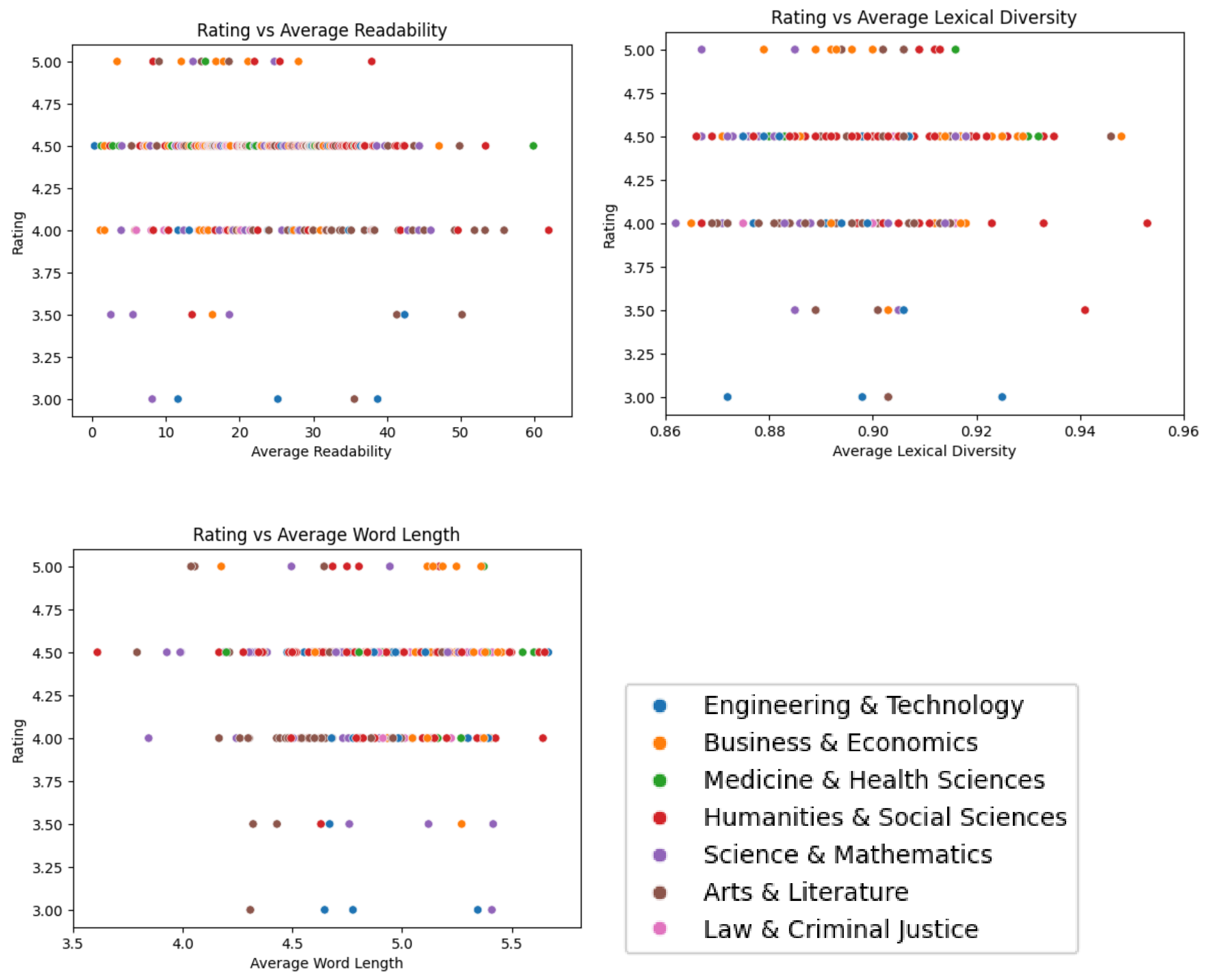




The pattern also holds true for this metric. Categories such as Medicine & Health Sciences are more challenging, while Arts & Literature and Sciences & Mathematics have more simple words. Arts & Literature have a p-value of 8.08e-08, and Medicine & Health Sciences have a p-value of 9.39e-08. Consequently, we conclude that categories as a whole have different levels of complexity compared to other categories.

```
Call:
lm(formula = avg_word_length ~ factor(category), data = df_cleaner)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.25998 -0.22856  0.01821  0.25540  0.84667 

Coefficients:
                                            Estimate Std. Error t value Pr(>|t|)    
(Intercept)                                  4.83702    0.01683 287.464  < 2e-16 ***
factor(category)Arts & Literature           -0.19342    0.03581  -5.402 8.08e-08 ***
factor(category)Business & Economics         0.15830    0.03997   3.961 7.96e-05 ***
factor(category)Engineering & Technology     0.02970    0.05053   0.588 0.556764    
factor(category)Humanities & Social Sciences  0.03430    0.03232   1.061 0.288801    
factor(category)Law & Criminal Justice       0.18131    0.04695   3.862 0.000119 ***
factor(category)Medicine & Health Sciences   0.27845    0.05181   5.374 9.39e-08 ***
factor(category)Science & Mathematics       -0.13641    0.03822  -3.569 0.000374 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3534 on 1095 degrees of freedom
Multiple R-squared:  0.1009,	Adjusted R-squared:  0.09519 
F-statistic: 17.56 on 7 and 1095 DF,  p-value: < 2.2e-16
```

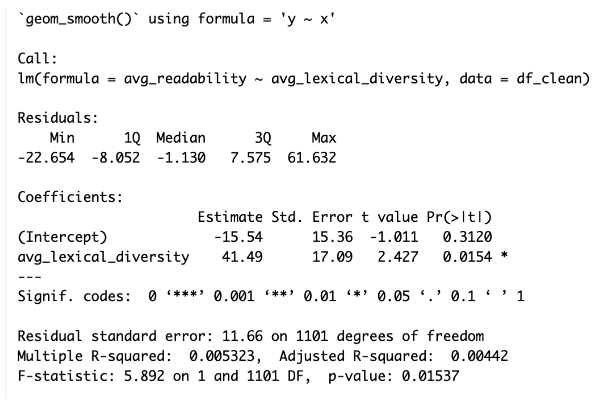
Next, we analyzed whether or not these metrics have an impact on a textbook’s rating. To do so, we create a graph for each metric with colors representing different categories. Additionally, we only plotted the textbook’s that had a rating (no textbooks had below a 3).



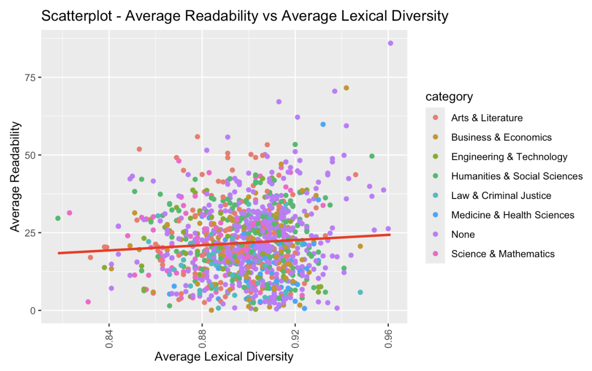
Identifying trends is difficult with the ratings having discrete values with a majority being either 4 or 4.5. As a result, no correlation was found between the difficulty metrics and the textbooks' ratings.

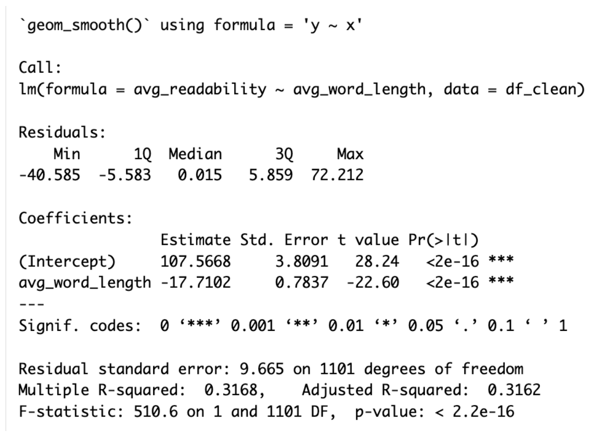
Finally, we analyzed how the metrics are correlated with each other. We measured how the average readability of each textbook is influenced by its lexical diversity and average word length. Readability and lexical diversity have a very weak positive correlation, 0.4149 readability score increase per 0.01 lexical diversity unit increase, with 0.0053 R^2. Also, the p-value of this analysis is 0.015. Thus, the correlation between them is significant.

In case of readability and word length, we found that a higher 0.1 average word length unit leads to a 1.77 average readability unit decrease, with 0.3168 R^2. This analysis indicates that there is a strong correlation between average readability and average word length. The p-value of this test is smaller than any reasonable significance level (0.005), so we are certain that this correlation is significant. The two analyses suggest that average word length has a stronger and more meaningful impact on readability than lexical diversity does.

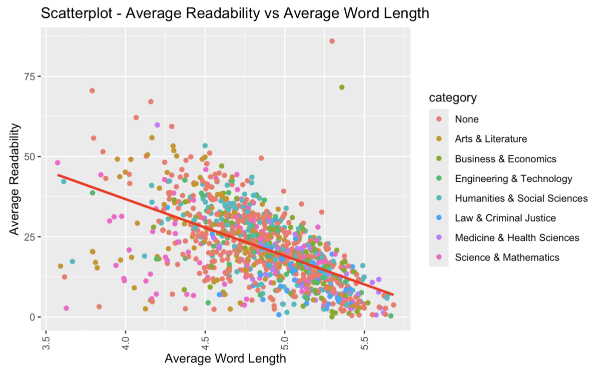


```
`geom_smooth()` using formula = 'y ~ x'

Call:
lm(formula = avg_readability ~ avg_lexical_diversity, data = df_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-22.654  -8.052  -1.130   7.575  61.632 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)  
(Intercept)             -15.54      15.36  -1.011   0.3120  
avg_lexical_diversity    41.49      17.09   2.427   0.0154 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.66 on 1101 degrees of freedom
Multiple R-squared:  0.005323,	Adjusted R-squared:  0.00442 
F-statistic: 5.892 on 1 and 1101 DF,  p-value: 0.01537
```



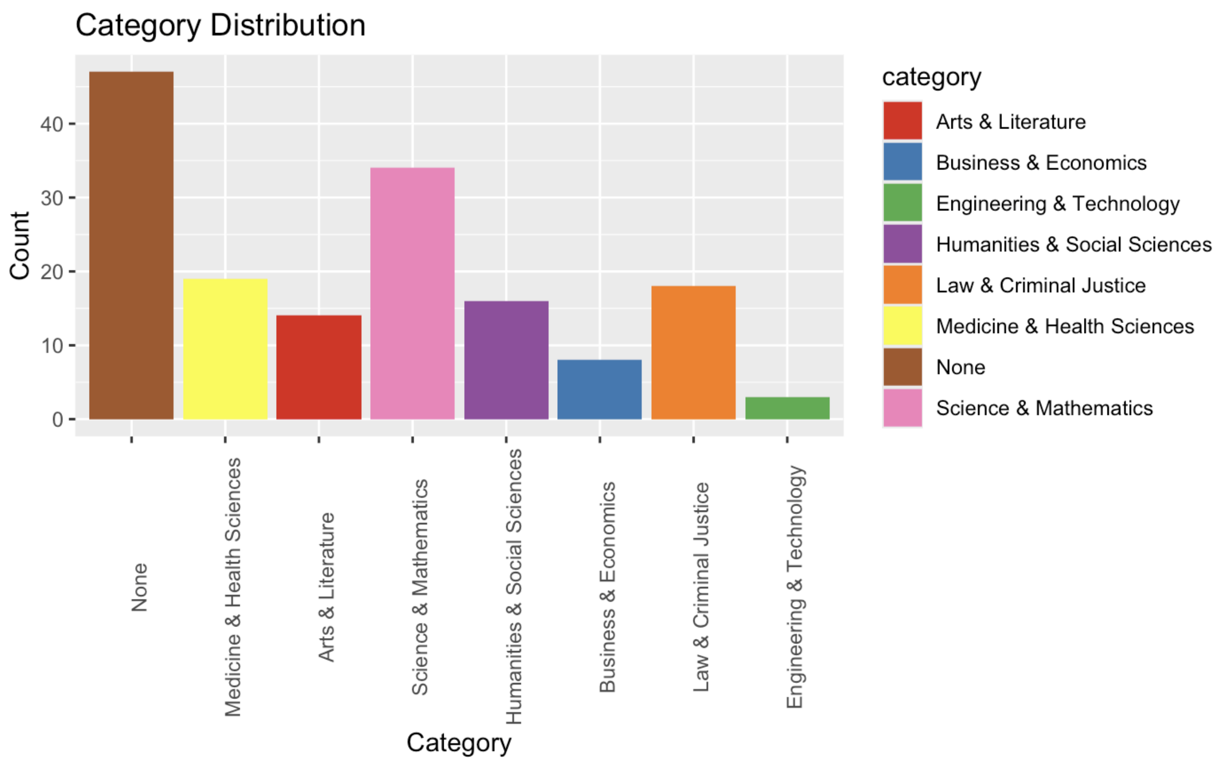


```
`geom_smooth()` using formula = 'y ~ x'

Call:
lm(formula = avg_readability ~ avg_word_length, data = df_clean)

Residuals:
    Min      1Q  Median      3Q     Max 
-40.585  -5.583   0.015   5.859  72.212 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)     107.5668     3.8091   28.24   <2e-16 ***
avg_word_length -17.7102     0.7837  -22.60   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.665 on 1101 degrees of freedom
Multiple R-squared:  0.3168,	Adjusted R-squared:  0.3162 
F-statistic: 510.6 on 1 and 1101 DF,  p-value: < 2.2e-16
```



8 Further Research

To continue this research project in the future, there are key areas of improvement that we can work upon. When performing exploratory data analysis and removing outliers from the created CSV file, a significant number of textbooks were found to have a negative Flesch Reading Ease score. While this score is possible using the formula, it did not make much sense and had to be examined further. To do so, each category's count of negative readability scores was graphed.



While all categories had some texts with negative scores, Science & Mathematics had a significant amount more. Normally, collegiate-level texts produce a Reading Ease score between 0 and 50. When examining "Fundamental of Matrix Algebra" by Gregory Hartman, it was found that its readability score was calculated to be -383, which indicates that there was something wrong.

We concluded that this issue was likely stemming from how our program categorized numbers and equations. To test this, a singular page from the textbook had its readability score computed. Using a page with eighteen exercises with multiple linear equations, we found that our program calculated a readability score of -204.16. This led us to conclude that it must be the equations that are causing faults.

Then, we made a slide adjustment to our code. In this new program, for a sentence to be considered, it must have at least three alphabetic words. Upon running this program on the same page from the last experiment, the new readability score was 45.63 – a much more logical value. When continuing this research, it will be important to implement this strategy and others to help prevent erroneous negative readability scores from being produced.

Additionally, different metrics can be used to see if they produce any correlations. For example, there is the Gunning fog index that calculates the reading level of text and takes ideas such as ‘complex’ words into consideration. Also, sentient analysis would be an interesting area to pursue. Seeing whether a text has a positive or negative tone and the resulting reviews could provide some insight.

Lastly, a dataset with more reviews and non-discrete scores would be crucial for identifying relationships between variables. For our dataset, some textbooks had to be dropped due to not having any reviews. For the ones that had reviews, the provided scores were discrete values either being 3, 3.5, 4, 4.5, or 5 stars out of 5. Having a continuous range of scores could better plot these relationships and visualize correlations.

9 Conclusion

To gain a deeper understanding on the design of quality textbooks throughout our research, we explored various Data Science such as data mining, natural language processing, and data visualization all while utilizing the High Performance Computing environment. Our data shows that there is no significant correlation between a textbook’s Flesch Reading Ease score, lexical diversity, or word length and the rating that the textbook receives. Even though we did not achieve our original goal, we did find some interesting conclusions. While all of the categories of textbooks had an average reading complexity that was considered to be at a collegiate level, some categories were found to be significantly more complex than others. For example, Medicine & Health Sciences textbooks were found to be consistently more complex across all three metrics that we evaluated. By improving our program, evaluating more metrics, and utilizing a more robust dataset, we hope to continue to gather more insight.

10 References

[1] “Flesch Reading Ease and the Flesch Kincaid Grade Level.” Readable. Published March 30, 2024. Accessed May 1, 2025. https://readable.com/readability/flesch-reading-ease-flesch-kincaid-grade-level/#:~:text=The%20Flesch%20Reading%20Ease%20gives,the%20average%20adult%20to%20read.