**TEXT MINING PROJECT**

**BOOK ANALYSIS “NO LONGER HUMAN” BY DAZAI OSAMU**

Huy Dang Nguyen

Graduate School of Business, Government, and Economics, Seattle Pacific University

ISM 6359: Data Mining

Professor Ryan LaBrie

March 2, 2023

**THESIS**

I recently read the book *No Longer Human* by Dazai Osamu and therefore wanting to perform a sentiment analysis on it given how depressing it is

**BOOK SUMMARY**

**Reference**

Dazai, O. (2020). *No longer human*. (D. Keene, Trans.). New Directions.

**Summary**

Here’s a quick summary of “No Longer Human” which is provided by its translator Donald Keene (1973) on the publisher’s website:

*Portraying himself as a failure, the protagonist of Osamu Dazai’s No Longer Human narrates a seemingly normal life even while he feels incapable of understanding human beings. Oba Yozo’s attempts to reconcile himself to the world around him begin in early childhood, continue through high school, where he becomes a ’clown” to mask his alienation, and eventually lead to a failed suicide attempt as an adult. Without sentimentality, he records the casual cruelties of life and its fleeting moments of human connection and tenderness. Semi-autobiographical, No Longer Human is the final completed work of one of Japan’s most important writers, Osamu Dazai (1909-1948). The novel has come to “echo the sentiments of youth” (Hiroshi Ando, The Mainichi Daily News) from post-war Japan to the postmodern society of technology. Still one of the ten bestselling books in Japan, No Longer Human is a powerful exploration of an individual’s alienation from society.*

**Reference:**

*No longer human*. New Directions Publishing. (1973, April 25). Retrieved March 2, 2023, from <https://www.ndbooks.com/book/no-longer-human/>

**TEXT RETRIEVAL**

**The Internet Archive**

The main benefit of book analysis is the ability to work with the rawest of raw data. I only need to import the novel onto RStudio and start the process of text mining instead of relying on another person’s ability to compile a good dataset. As it is simply unwise and a waste of time to manually type up my physical copy of the book, I resorted to finding an online full-text version of the book for ease of import; the plan is to copy-paste the raw text onto Notepad, iron out the kinks and then import it to RStudio.

Fortunately, the Internet Archive has a copy of the book online with a raw text option. Unfortunately, the formatting of the raw text is a complete mess. Here are a couple of examples:

Text

Description automatically generated

*Figure 1: Word Typo. Retrieved from the Internet Archive.* [*https://archive.org/stream/no-longer-human/no%20longer%20human\_djvu.txt*](https://archive.org/stream/no-longer-human/no%20longer%20human_djvu.txt)

Text

Description automatically generated

*Figure 2: Random paragraph cutoff. Retrieved from the Internet Archive.* [*https://archive.org/stream/no-longer-human/no%20longer%20human\_djvu.txt*](https://archive.org/stream/no-longer-human/no%20longer%20human_djvu.txt)

Text

Description automatically generated

*Figure 3: Weird random character. Retrieved from the Internet Archive.* [*https://archive.org/stream/no-longer-human/no%20longer%20human\_djvu.txt*](https://archive.org/stream/no-longer-human/no%20longer%20human_djvu.txt)

I gave up on this text after quickly skimming through it. There were just too many errors to fix that it would just be a complete waste of my time, especially considering the tight deadlines.

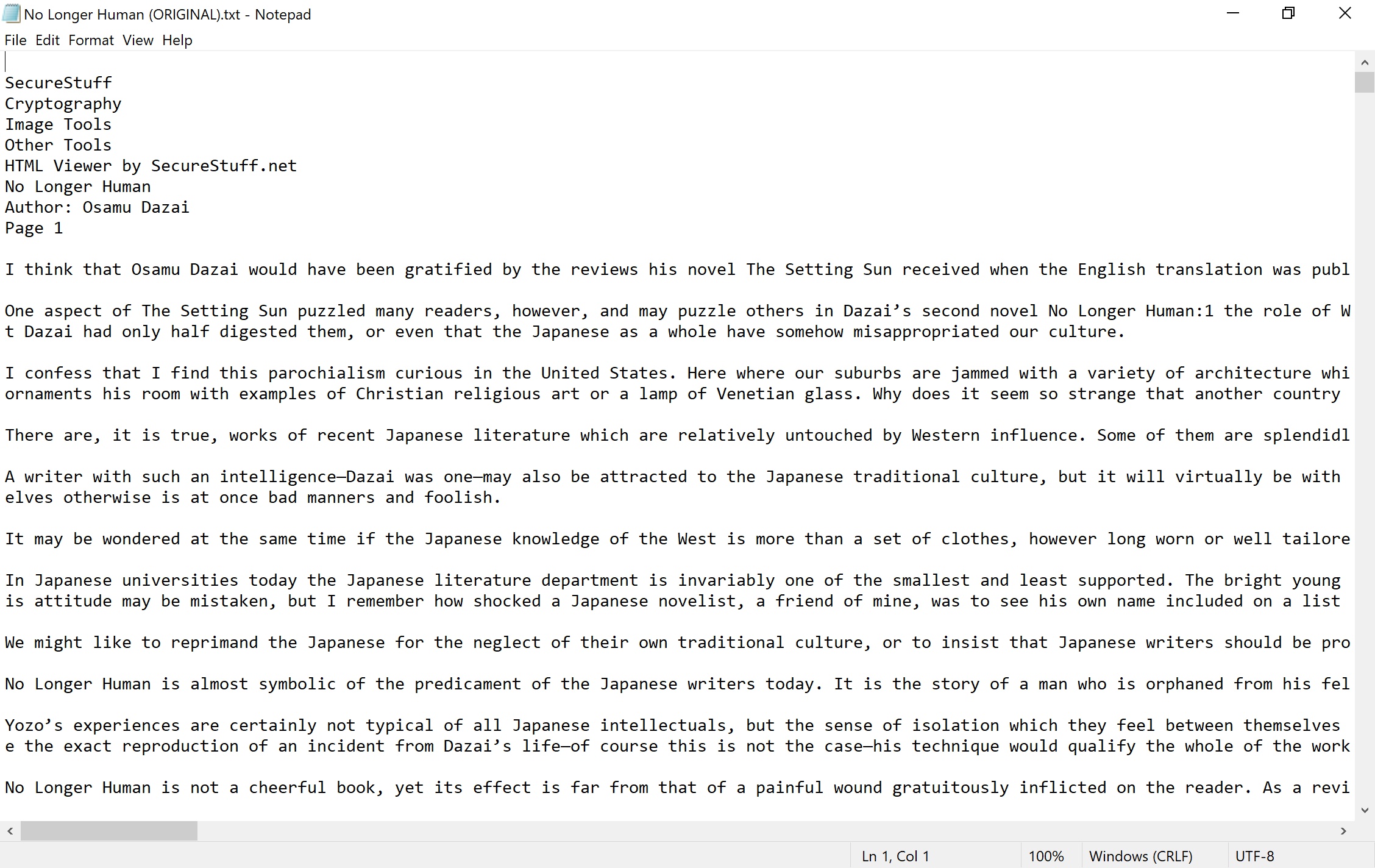
**Reference:**

*No longer human: Osamu Dazai: Free Download, borrow, and streaming*. Internet Archive. (1970, January 1). Retrieved March 2, 2023, from <https://archive.org/details/no-longer-human/>

**All Free Novel**

I managed to find another raw text copy of the book on a random site called “All Free Novel.” The process of retrieving the book was quite sketchy as the text file was encrypted, thus, required a “security key” to access. Even then, one can only read it on the browser. Luckily, I did not encounter any problem copy-pasting the text onto Notepad.

The format of this text was simply superior to that of the Internet Archive. That said, some cleaning is in order before I import it into RStudio. Here’s an excerpt of the original text in Notepad:



*Figure 4: Notepad view of the book’s raw text*

**Reference:**

*Read no longer human by Osamu Dazai Online Free*. AllFreeNovel. (n.d.). Retrieved March 2, 2023, from <https://www.allfreenovel.com/Book/Details/55235/No-Longer-Human>

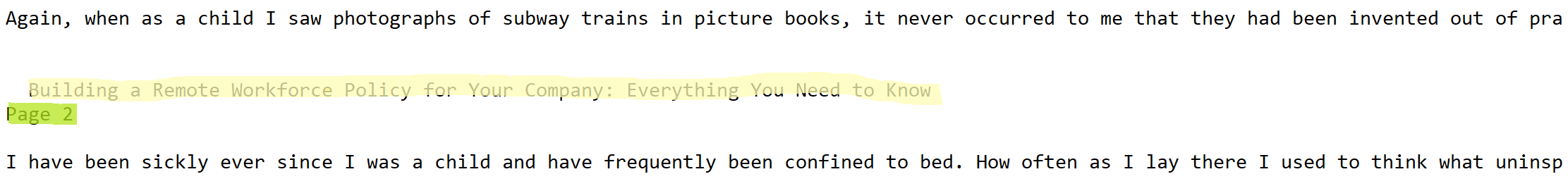
**REFORMATTING TEXT (PRE-RSTUDIO)**

**Tool**

For reformatting the .txt file, I prefer Notepad++ over Notepad as it has more features for search and replace.

**Removing unrelated texts**

Somehow, the webpage advertisements got mixed up in the txt file during the copy-paste process. Fortunately, they only appear at the end of each page number (a unique identifier of this website). Thus, I removed all of the advertisements and the page numbers via Ctrl + F.



*Figure 5: Advertisement texts (yellow) before a page number (green)*

**Re-Coding Chapter Name**

The code I used to extract chapter number require a numeric chapter system (e.g., Chapter 1) to work properly. This book uses a fancy chapter system (e.g., The First Notebook), which ties to the plot of the book. Therefore, I converted all chapter names to numeric, including the prologue and epilogue. I realized now that I could have merged the two parts of chapter 3 in the txt file (removing the title for part II and renaming part I to just chapter 3) instead of merging them in RStudio.

dat2 🡨 dat %>%

mutate(linenumber = row\_number(),

chapter = cumsum(str\_detect(text,

regex("^chapter [\\divxlc]",

ignore\_case = TRUE))))

*Code Block 1: Sample (edited) code for extracting line number and chapter. Using the cumulative sum of each instant of the word chapter (find via a regular expression, case insensitive) in the document.*

**Reformatting Line Structure**

Each line of the text is an entire paragraph. While this is nice, I wanted each sentence to have its line for later analysis of sentiment flow. A naïve approach to this tedious task would be to Ctrl + F with the keyword being “.” to find the end of each sentence. In contrast, sentences can end with non-period symbols as well (i.e., “?”, “!”).

A better way of doing this would be to find all capitalized words in the text. While not perfect as there are many other reasons for words to be capitalized (e.g., names, etc.), it is better to filter those out than to find “?” and “!”. To achieve this, I used the “mark all” function of Notepad++ with the specific settings:

* Find what: \b[A-Z]\w+ (regex for finding capitalized words)
* Enables “Match Case”
* Enables “Wrap Around”

The weakness of this approach is that some capitalized words are not highlighted (e.g., “I”). Thus, I still had to come in and scan through the document to make sure that everything is correct. I kept everything between “()” and “” “” on the same line just in case some analysis required it which I later found out not to be the case. Despite the tedium, I got the opportunity to read the book again so it was a semi-enjoyable process.

Text

Description automatically generated

*Figure 6: Notepad++’s mark setting and output*

**Deep Cleaning**

The last step before importing the txt file into RStudio is to iron out all of the minor errors. There’s no easy way to do this so I manually went through the txt file to fix them while reformatting the sentence structure.

**TEXT MINING**

**Libraries**

Here are all the libraries that I used in this project.

* Tidyverse
* Stringr
* Scales
* Tidytext
* Textstem
* Export
* AFINN

**Importing Data**

I used the “read.delim” (read delimited file) to import the text onto RStudio, the process was surprisingly smooth and easy. Since I set “header = F” to keep all my rows, the attribute name is now “V1.” I then changed it to “text” by creating a copy of a column name “text” and deleting the “V1” column.

dat 🡨 read.delim("No Longer Human.txt", header = F)

dat$text 🡨 dat$V1

dat 🡨 subset(dat, select = -c(V1))

*Code Block 2: Importing Text onto RStudio*

**Data Prepping**

After extracting the line number and chapter to be separate attributes (Code Block 1) and merging the two parts of chapter 3, I perform the basic text mining prepping: tokenizing 🡪 converting string to lower case 🡪 lemmatization 🡪 removing stop-words (RStudio default stop-words library). My way of prepping data has the word coded as records instead of attributes. Therefore, I can only track the observation/word count after each data-prepping step was applied. I started with 4857 observations and ended up with 3443 observations after prepping.

tidyDat 🡨 dat2 %>% unnest\_tokens(word,text)

*Code Block 3: Tokenization. Word Count: 4857*

tidyDat$word 🡨 str\_to\_lower(tidyDat$word, locale = "en")

*Code Block 4: Convert String to Lowercase. Word Count: 4857*

tidyDat$word <- lemmatize\_words(tidyDat$word)

*Code Block 5: Lemmatization. Word Count: 3798*

data("stop\_words") #Default stop-words library

customStopWords 🡨 bind\_rows(tibble(word = c("i’m", "it’s"),

lexicon = c("custom")),

stop\_words) #Add custom stop-words

tidyDat 🡨 tidyDat %>% anti\_join(customStopWords) #Removing stop-words

*Code Block 6: Removing Stop-words. Word Count: 3443*

**Word Frequency**

I wanted to create several graphs that display words with the highest frequency in the book and for each chapter with sentiment labeled on the appropriate one.

**Data Preparation**

First, I imported the AFINN sentiment. I chose this specific library because the sentiments are on a scale from very negative (-3) to very positive (+3) compared to just being labeled “positive” or “negative” in the NRC and BING libraries. NRC does have the advantage of measuring sentiment on different emotions (e.g., disappointment, anger, joy, etc.) though that might be too out of scope for this project.

I then merged the two datasets, keeping the text dataset (meaning the sentiment column will display NA if the word is not in the library). Additionally, I also labeled the sentiments for graphing purposes (> 0 is positive, 0 < is negative, and = 0 is neutral). Since there’s no neutral word in this book, I filter out the single neutral word in the AFINN dataset. Lastly, I sort the dataset by the word count in descending order. We are now ready for graphing!

AFINN <- get\_sentiments("afinn") #Importing AFINN Sentiment Library

AFINN$Sentiment[AFINN$value < 0] 🡨 "-" #Negative Sentiment

AFINN$Sentiment[AFINN$value == 0] 🡨 "/" #Neutral Sentiment

AFINN$Sentiment[AFINN$value > 0] 🡨 "+" #Positve Sentiment

AFINN <- AFINN %>% filter(Sentiment != "/") #Filter out 1 neutral word

*Code Block 7: Importing and labeling AFINN sentiments*

datCount 🡨 tidyDat %>% count(word, sort = T) %>%

merge(., AFINN, by.x = "word", by.y = "word", all.x = T) %>%

arrange(desc(n)) %>%

mutate(word = reorder(word, n)) %>%

head(., 20)

*Code Block 8: Prepping Data for Graphing Word Frequency (Entire Book)*

**Graphing Results**

I used ggplot to make all graphs in RStudio. Since all word frequency graphs and their prepping process are very similar structurally, I will only show one as an example.

ggplot(datCount, aes(n, word))+ #Graph grids

theme\_classic()+ #Graph theme

geom\_col(aes(fill = n), show.legend = F)+ #Plot type

geom\_text(aes(label = Sentiment), nudge\_x = 3)+ #Adding sentiment label

scale\_x\_continuous(name = "Word Count")+ #Renaming the x-axis

theme(plot.title = element\_text(face = "bold"))+ #Bold Title

labs(title = "Word Frequency Across the Book", #Plot title

caption = "'+' Indicates Positive Sentiment, '-' Indicates Negative Sentiment", #Plot caption

y = NULL) #No name for the y-axis

*Code Block 9: Bar Chart Code Breakdown*

Here are the graphs of the word frequency of the entire book and by each chapter:

Chart

Description automatically generated

The top five words with the highest frequency across the book are (1) feel, (2) time, (3) woman, (4) human, and (5) horiki. Looking up each iteration of feel and time, I don’t find them as significant as this is a semi-autobiography book that is written from a first-person perspective. Therefore, it is normal that Yozo, the protagonist, would discuss his feelings. Furthermore, this is also a coming-of-age story so it is unsurprising that “time” is brought up multiple times.

On the contrary, the next three words (1) woman, (2) human, and s(3) horiki are more interesting.

**Woman**

A large portion of this book is dedicated to Yozo’s relationship with the women throughout his life. Despite his negative view towards women, he is a textbook-lady man who is over-dependent on them for shelter and money which he would waste on boozes and tobacco. In extreme cases, he manipulated them into committing double suicide with him. Interestingly, Yozo’s character-defining moments would inevitably involve a woman in one way or another, for better or worse.

**Human**

For one to understand what disqualifies Yozo from being human (the literal translation of this book’s title), one must also understand what makes one a human being. Consequently, it is not that Yozo, and the author by extension, does not understand how to be human as he wouldn’t be able to write this book at all. It is the fact that he cannot experience and therefore relate to the experience of being a normal human being that disqualified him from being human.

**Horiki**

Horiki, an acquaintance of Yozo, is a recurring character throughout the book. While neither of them thinks highly of the other person, they do appear to enjoy each other company on some level. After all, Horiki is one of the few male friends that Yozo has. It is an interesting relationship.

**Sentiment Words**

“Leave” and “smile” are sentiment words with the former having a negative connotation and the latter having a positive connotation. In the context of this book, however, both are misrepresented. In fact, the only reason “leave” even made it to the top 20 was because half of it refer to tree leaves as opposed to the act of leaving.

On the other hand, “smile” should signify negative sentiment instead of positive as in the context of the book, the protagonist uses his fake smile as a way to blend in with normal people.

Chart, funnel chart

Description automatically generated

Prologue mainly discuss an outsider’s view of each of the stage of the life of the protagonist, represented through each chapter of the book. A lot of emphasis was put on facial expressions, with the highlight being consistently the fake smile, which was pointed out earlier as having a negative sentiment instead of a positive one.

Chart, bar chart

Description automatically generated

Chapter 1 focuses on the Yozo recount of his childhood with the main focus being the process of learning to please his family, especially his father. The word “laugh” here does signify positive sentiment as it refers to Yozo’s dad laughing instead of Yozo.

Chart, bar chart

Description automatically generated

Chapter 2 is about the high school and university life of Yozo, this is where discussion of women skyrocketed. One could also see that there are a lot of mentions of other important characters: (1) Horiki – Yozo’s (at this point) university friend, (2) Takeichi – the first person to see through Yozo’s “clowning,” and (3) Tsuneko – Yozo first “lover” who commit suicide with him.

Chart

Description automatically generated

As the two parts of chapter 3 were merged into one, the top 10 appear to be keywords of various story importance moments (e.g., the relationship between Horiki and Yozo, Yozo’s drinking problem after becoming a mainstream artist, the sexual assault of Yoshiko, etc.).

The only new sentiment word here is “crime” which most of its usage was in a game between Horiki and Yozo where they attempt to find an antonym for the word. The game itself was difficult to follow due to the loss of meaning in translation.

Chart, funnel chart

Description automatically generated

The epilogue does back to the outsider view mentioned in the prologue. The main topic again is the notebooks that contain the story of the novel. Other than that, nothing else significant is worth mentioning.

**Sentiment Analysis**

**Data Preparation**

I attempted to capture the sentimental flow of the overarching narrative of the book. To do this, I first inner join the data set with the AFINN sentiment library to keep only sentiment words. Next, I created an index column to capture each narrative “block” of the book where index = line number %/% (integral division) 20. Lastly, I convert the table to a pivot table with sentiment score = count of positive sentiment – count of negative sentiment in each index.

*Note: It is not worthing that 20 is kind of an arbitrary number. The guide I follow divides 80 for their book which has around 120k words while mine only has 30k. Therefore, I did a simple conversion and got said number 20.*

narrativeSent 🡨 tidyDat %>%

inner\_join(AFINN) %>%

count(chapter, index = linenumber %/% 20, Sentiment) %>%

pivot\_wider(names\_from = Sentiment, values\_from = n, values\_fill = 0) %>%

mutate(Sentiment = `+` - `-`)

*Code Block 10: Preparing data for Sentiment Analysis*

**Graphing Results**

Chart, line chart

Description automatically generated

Overall, the count of negative sentiments overwhelmed that of positive sentiments with massive spikes throughout the narrative of this book.

Timeline

Description automatically generated  
Here’s the same graph but grouped by chapters. Due to time constraint, I was not able to examine each peak to see if it’s reflective of the narrative structure. However, the validity of these graphs come into question when examining the contributing sentiment words

Chart, bar chart

Description automatically generated

Despite having the highest contribution to the sentiment score, both “leave” and “smile” are wrongly coded, with half of the count of the former referring to a plant structure and the latter having negative sentiment in the context of the book. “Smile” specifically cast doubt on many positive sentiment words as this book is about a person who desires but couldn’t be “human”. Therefore, many of these words might have a negative sentiment instead of positive sentiment (i.e., “forgive” refer to the act of begging for forgiveness rather than forgiving someone). In contrast, the negative sentiment column looks good to me.

Chart, pie chart

Description automatically generated

Mapping the percentage of distinct sentiment words in each category, it was surprising to see that the difference between positive and negative was a mere 3.8%. That said, the true difference is probably larger given that I didn’t modify the sentiment library to the context of this book. I have also included a table of sentiment percentage groups by chapter (figure 7).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chapter** | **Positive Words** | **Negative Words** | **Total Words** | **Negative Percentage** | **Positive Percentage** |
| Prologue | 18 | 22 | 301 | 7.31 | 5.98 |
| 1 | 86 | 137 | 1332 | 10.29 | 6.46 |
| 2 | 259 | 420 | 4296 | 9.78 | 6.03 |
| 3 | 324 | 543 | 5253 | 10.34 | 6.17 |
| Epilogue | 17 | 18 | 288 | 6.25 | 5.90 |

*Figure 7: Sentiment Ratio by Chapter*

**TF-IDF**

I performed TF-IDF analysis mainly to compare and contrast the results to the results of Word Frequency. Overall, I think this book is too short for this kind of analysis. Therefore, I would not put too much weight on the results

The code for TF-IDF was surprisingly easy to write (Code Block 11)

TF\_IDF 🡨 tidyDat %>%

bind\_tf\_idf(word, chapter, n) %>%

arrange(desc(tf\_idf))

*Code Block 11: TF-IDF by Chapter*

**Graphing Result**

Chart, bar chart

Description automatically generated

Despite my previous concerns after hearing you, professor LaBrie, discussing TF-IDF weakness when applying to small documents, it was able to highlight a couple of significant words in each of the chapters, especially the longer ones (i.e., chapters 2 and 3). On the contrary, the words that are outside of the top 3-5 appear to be all over the place. I wonder if there’s a way to optimize this function.

**Bi-Gram Analysis**

The code for bi-gram analysis is quite simple, it is exactly the same as tokenization but with a minor change in the parameter with “token = ‘ngrams’” and specifying that “n = 2” (Code Block 12)

datBi 🡨 dat %>%

unnest\_tokens(bigram, text, token = "ngrams", n = 2) %>%

filter(!is.na(bigram))

*Code Block 12: Tokenizing Bi-Gram*

**Chart

Description automatically generated**

As you can see on the x-axis of this graph, the highest bi-gram count is 8. Therefore, I couldn’t do any further analysis. I did attempt to perform a TF-IDF but the graphs were identical to that of word frequency. That said, given the range of bi-gram frequency, the author may repeat them with specific intentions.

One bigram that captures my attention is “immaculate trustfulness,” this word is used to describe the character Yoshiko’s naivety. This “immaculate trustfulness” of the protagonist’s wife led to her being sexually assaulted by a local shop owner. The repetition of this word represents the struggle of the protagonist in coping with the situation: “Is immaculate trustfulness after all a

source of sin?” he wondered.

**Reflections**

**What Went Well**

Overall, I think this project went better than I expected, mostly thanks to Julia Silge & David Robinson. If it was not for their book “Text Mining with R – A Tidy Approach,” I wouldn’t be able to make such nice graphs and analyses. Furthermore, working with the raw text of a book reduces the risk of developing commitment issues compared to working with datasets, where one attempts to find the perfect dataset. Lastly, the functions that text mining uses are way more forgiving than that of data mining. I was able to spend my time analyzing the results instead of troubleshooting syntax/logic errors.

**What Did Not Go Well**

On the other hand, text mining is extremely context-dependent. This is shown most clearly in my sentiment analysis where many of the default positive sentiment words did not apply as the plot was about a person who desires but is unable to achieve those positive traits (i.e., “smile” in the context of a book is an act of clowning in an attempt to fit in). Furthermore, sentiment is also subjective. In the example above, one could also argue that “smile” is still positive as it enables Yozo to fit in with the rest of society on some level.

Another problem I encounter was with TF-IDF and bi-gram as my book was too short for those analyses. In their book, Silge & Robinson was analyzing six of Jane Austen’s novels whose word count totaled up to 600,000 compared to a mere 30,000 words in my book. Consequently, I was only able to perform basic analysis.

The last concern is the book itself. I chose this book because it is a uniquely depressing book that piques my interest recently. However, during the process of this write-up, I realize that the content of it may not be suitable for a presentation. Therefore, I might either put a trigger warning or just avoid all of the sensitive topics altogether.

**What Would I Do Differently Next Time**

First and most importantly, I would extract the top 10-20 values of word frequency for each chapter, go back into the book for the context and recode its sentiment if necessary. This would ensure the validity of my graphs as opposed to the current ones which look pretty but do not quite do their job well.

Secondly, I might increase the scope from just one book to two-three books by the same author to be able to perform the deeper level of TF-IDF and N-Gram analyses. Furthermore, I can compare and contrast these works to see whether said author sticks to one theme or not.

Lastly, I might want to look into implementing a word cloud diagram. I saw an absolutely beautiful example in Silge & Robinson’s book and want to replicate it. However, knowing my perfectionism, I limited myself to simple graphs to meet the deadlines.

**Reference**

*Welcome to Text Mining with R* by Julia Silge & David Robinson (ND) Tidy Text Mining. Retrieved March 6, 2023, from <https://www.tidytextmining.com/index.html>