**Part B**

By Ismail Hendryx

**Introduction:**

For Part B of the assignment, I underwent several types of analysis including processing, sentence analysis, frequency distribution, data visualisation & word analysis.

**Are there additional pre-processing steps you could perform on data that might be helpful?**

Looking at this dataframe I noticed several areas that could have been cleaned before being experimented with. For instance, during PART A I had to use the .lower() function above when generating Lexical Richness. This was because when comparing tokens such as ‘Thank’ to ‘thank’, they would be considered different tokens. Additionally, adding a column containing the tokens without the stopwords would be beneficial to use.

**PART B 1.1 Using .lower():**

**Method:**

Here we convert all tokens in the 'Tokens' column to lowercase. It creates a new Series named `lowered\_Tokens` by applying a lambda function to each token list, converting each token to lowercase. Finally, the modified Series `lowered\_Tokens` replaces the original 'Tokens' column in `df`.

**PART B 1.11 Removing stopwords:**

**Method:**

This code snippet employs list comprehension and the `assign()` method to create a new column named 'TokenKeywords'. It iterates through each row in the 'Tokens' column of the DataFrame, filtering out predefined stopwords from the token lists. The resulting 'TokenKeywords' column contains refined token lists, enhancing the data's focus on informative keywords for further analysis or modeling.

**Did I notice any patterns during Part A and what can we learn if we group the data together in different ways? Does this give us more useful information?**

**PART B 1.2 Grouping Full Stops with Status**

In Part A of the assignment, I observed a substantial difference in the average token length between rejection and non-rejection emails, with rejection emails having an average length of over 25 tokens shorter than non-rejection emails (106.22 compared to 80.31 tokens). This discrepancy prompted me to further investigate the average number of sentences per email for rejection emails compared to non-rejection emails. While this method isn't foolproof since full stops may appear in contexts other than ending sentences, it presents an intriguing avenue for exploration. By counting the number of full stops in the token sequences, I aim to gain insights into potential differences in sentence structure and complexity between rejection and non-rejection emails.

**Method:**

This code snippet adds a new column named 'NumFullStops'. It achieves this by applying a lambda function to the 'Tokens' column of the DataFrame, which counts the number of full stops ('.') in each token sequence. The result is stored in the new 'NumFullStops' column, providing a count of the full stops present in each email.

Screens screenshot of a computer

Description automatically generated

**Results:**

My results showed that the average length for not rejected emails was 4.30 but for rejected it was 3.63. This matches the earlier findings in Part A where we found that the number of tokens for rejection emails was significantly lower. This suggests that rejection emails are shorter than normal emails.

**How do the frequency distributions compare for rejection emails compared to non-rejection emails? Any interesting observations?**

**PART B 1.3 Frequency distributions**

Below I visualise the frequency distribution of token counts in rejection and non-rejection emails.

**Method:**

In the code we begin by filtering the DataFrame into two subsets: one containing only rejection emails ('reject\_emails') and the other containing only non-rejection emails ('non\_reject\_emails'). Then, using Matplotlib, histograms are plotted for each subset, with the x-axis representing the number of tokens and the y-axis representing the frequency of emails with a particular token count. The histograms are plotted with 30 bins and transparency set to distinguish between rejection and non-rejection email distributions. The resulting visualisation allows for a comparative analysis of the token count distributions between rejection and non-rejection emails, providing insights into potential differences in email length and complexity between the two categories.

A screenshot of a computer

Description automatically generated

**Results:**

The majority of the tokens for rejections are less than for the non-rejections.

**What happens if you conduct sentiment analysis?**

**PART B 1.4 Word analysis**

Having applied to a lot of roles myself with a lot of rejections, a common theme I noticed was the use of apologetic words. Words such as ‘Sorry’, ‘Unfortunately’ & ‘Regret’ seemed prominent from my experience. Alongside using ChatGPT to ask what it thinks the most common apologetic words were, I made a list of apologetic words ‘chatGPT\_Apologetic’.

**Method:**

This code initialises a counter `j` to tally occurrences of apologetic words in a dataset. It iterates through each apologetic word, normalising capitalisation for accurate matching. Then, it traverses each token in the dataset, normalising its capitalisation as well, checking for matches with apologetic words, and updating `j` accordingly. The frequency of apologetic words is determined by the total count in `j. In this dataset, all 56 occurrences of apologetic words were found in rejection emails. Notably, 86.2% of the rejected emails contained at least one apologetic word.

**Results:**

Upon reviewing both rejection and non-rejection emails, it's evident that all 56 instances of apologetic words occurred exclusively in the rejection emails. This finding strongly suggests a correlation between the apologetic words and rejection emails. To further explore this relationship, we can analyze the proportion of rejection emails containing at least one apologetic word from the `chatGPT\_Apologetic` set. If a notable percentage of rejection emails include these words, it could imply that detecting apologetic language might effectively identify rejection emails. This approach holds potential for automating the identification of rejection emails in large datasets, streamlining email processing, and enabling prompt responses or follow-ups.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

**Word Analysis 1.41**

**PART B Looking for the 10 most popular words in rejection emails**

Another exploration I underwent was looking for the 10 most popular words in rejection emails and then seeing how many times they appeared in both rejection and not\_rejection emails.

**Method:**

This snippet of code performs a group-by operation on the Dataframe based on the 'Status' column, which indicates whether an email is a rejection or non-rejection email. The `groupby()` function is used to group the data by the values in the 'Status' column, and then the `sum()` function is applied to concatenate the lists of tokens within each group. This results in a Series where the index corresponds to the unique values in the 'Status' column ('reject' and 'not\_reject'), and the values represent the combined token lists for each group. Finally, the token lists for rejection and non-rejection emails are extracted separately from the resulting Series, assigning them to the variables `reject\_tokens` and `non\_reject\_tokens’. Most importantly, we also use TokenKeywords instead of Tokens to ensure the top words weren’t stopwords.

Then using a function named `get\_top\_ten\_common\_words`, we take a list of tokens as input. Inside the function, where it uses the `Counter` class to count the occurrences of each token in the input list. Then, it calls the `most\_common()` method on the `Counter` object to retrieve the ten most common tokens and their corresponding counts. Finally, the function returns these top ten tokens and their counts as a list of tuples. The code then applies this function to the token lists of rejection and non-rejection emails (`reject\_tokens` and `non\_reject\_tokens`), to find the top ten most common words in each category. It prints out the top ten most common words for rejection emails along with their frequencies.

**Results:**

We can see below that the 10 most popular words arent mentioned as much in non rejection emails as in rejection emails. From this we can possibly infer that reject emails have certain keywords that arent mentioned

A screenshot of a computer

Description automatically generated A screenshot of a computer program

Description automatically generated

**Word Analysis 1.42**

**PART B Sentiment Analyser**

In this section I analyse the words in all the emails hoping to find a pattern.

**Method:**

This code analyses the sentiment of words in rejection and non-rejection emails and then plots the top 10 most common positive and negative words separately for both types of emails in four bar charts.

**Results:**

In analysing sentiment, it's unsurprising to find few negative words in rejection emails since professional communication typically avoids emotive language. However, there were numerous positive words in rejection emails, suggesting a need for empathy and encouragement, although this wasn't explored further as similar positive language was also present in non-rejection emails.

A screenshot of a computer

Description automatically generated

**Word Analysis 1.43**

**Exploring the Phrase 'Thank You'**

Noticing the pattern of the phrase ‘Thank you’ being in reject emails I decided to explore how often this was mentioned in all rejection emails.

**Method:**

This code converts tokens to lowercase, filters rejection emails with "thank you", and displays samples.

**Results:**

We can see that 81.54% of rejection emails have 'Thank you' in them suggesting this phrase is used in the majority of reject emails.

Additionally, we recognise even further that this is a strong pattern because in not\_reject emails the phrase only occurs 32.81% of the time which is much lower.

A screenshot of a computer

Description automatically generated

**Conclusion:**

Overall, during Part B’s analysis I delved into various aspects, including pre-processing, sentiment analysis, and word frequency distribution. Notable patterns emerged, such as the prevalence of apologetic language in rejection emails and the significant occurrence of the phrase "Thank you," suggesting distinct communication styles between rejection and non-rejection emails.