NLP Project Report- Round1

Submitted by

Team ALP

Hemang Goyal-19ucs193

Link to GitHub code repository -

https://github.com/Hemang110/NLP_Project

Problem Description -

We were tasked to take two books from http://gutenburg.org in text format and perform Natural language processing techniques on the text acquired.

- Data pre-processing
- Analyze the frequency distribution of tokens in book1 and book2 separately
- Create a Word Cloud of book1 and book2 using the token that you have got
- Evaluating relationship between word length and frequency for book1 and book2
- Parts of Speech tagging for the words in the text

Books used-

T1: Animal Life in Field and Garden by Florence Constable Bicknell and Jean-Henri Fabre

Link - https://www.gutenberg.org/ebooks/66755

T2: David Vallory by Francis Lynde

Link- https://www.gutenberg.org/ebooks/66754

Python Libraries/Modules used

Matplotlib : Creates a figure, Creates a plotting area in a figure

re library (regular expressions library): Provides support for regular expressions

nltk: : Used in tokenizing, removing stop words and pos tagging

Math : For calculating floor and ceil values

WordCloud : For creating word cloud

Collections : For getting the frequency mappings of the POS tags

In [3]:
```
import sys
print(sys.executable)
```

```
C:\Users\91916\anaconda3\python.exe
```

```
# Importing libraries
```

In [23]:
```
import re
import nltk
nltk.download('punkt')
nltk.download('wordnet')
import math
import numpy as np
from wordcloud import WordCloud
import matplotlib.pyplot as plt
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\91916\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\91916\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Inference from Raw text-

The Raw text contains Chapter number and headings, indexes, unprocessed data which is not fit for processing.

Data Pre-processing steps-

Performed following data pre-processing steps -

1. Used Regular expressions to remove chapter number and chapter Headings.
2. Removed all punctuation marks using regular expressions.
3. Changed text to lowercase.
4. Conserved the meaning of short forms like can't, won't, 'm, 're etc. by converting short forms to actual representations.
5. Tokenized the text into a list of words.
6. Removed unnecessary data.
7. Removed hyperlinks

Preprocessing snippets-

Preprocessing

In [62]:
```
#reading text from the books
T1= open("C:\\Users\\91916\\Desktop\\Animal Life in Field and Garden.txt",encoding="utf-8").read()

T2=open("C:\\Users\\91916\\Desktop\\DAVID VALLORY.txt",encoding="utf-8").read()
```

In [70]:
```
#remove unneccesary text
def remove_text(text):
    text = re.sub(r'\*\*\* START OF THE PROJECT ([\s\S]*?)\*\*\*', ' ', text)
    text = re.sub(r'\n\*\*\* END OF THE PROJECT ([\s\S]*)',' ',text)
    text = re.sub(r'\n+CHAPTER.*\n\n.*\n+','\n\n', text)
    text = re.sub(r'\n+^M{0,4}(CM|CD|D?C{0,3})(XC|XL|L?X{0,3})(IX|IV|V?I{0,3})$\n.*\n','\n\n', text)
    return text
```

In [72]:
```
T1=remove_text(T1)
T2= remove_text(T2)
```

In [73]:
```
# convert all text to lower case and removing any link
def to_lower(text):
    text = text.lower()
    re.sub(r"http\S+", "\n", text)
    return text
```

In [75]:
```
#convert short forms to full forms
def convert(text):

    text = re.sub(r"won\'t", "will not", text)
    text = re.sub(r"can\'t", "can not", text)
    text = re.sub(r"n\'t", " not", text)
    text = re.sub(r"\'re", " are", text)
    text = re.sub(r"\'s", " is", text)
    text = re.sub(r"\'d", " would", text)
    text = re.sub(r"\'ll", " will", text)
    text = re.sub(r"\'t", " not", text)
    text = re.sub(r"\'ve", " have", text)
    text = re.sub(r"\'m", " am", text)
    return text
```

In [77]:
```
T1 =to_lower(T1)
T2=to_lower(T2)
T1 = convert(T1)
T2= convert(T2)
```

In [78]:
```
#removing punctuation
T1=re.sub(r'[^\w\s]','',T1)
T2=re.sub(r'[^\w\s]','',T2)
```

Tokenization snippets-

```
# Tokenisation
```

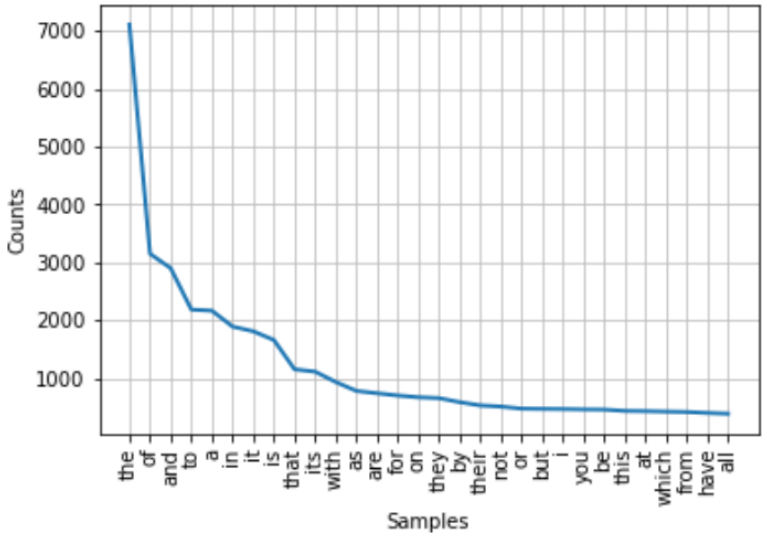
In [79]:
```
from nltk.tokenize import word_tokenize

# splitting text into individual tokens(words)
def tokenize_word(text):
    words = word_tokenize(text)
    return words
```

In [80]:
```
words_T1=tokenize_word(T1)
words_T2=tokenize_word(T2)
```

Frequency Analysis Snippet-

Frequncy Analysis

In [81]:
```
# analyzing frequency of words before removing stop words
frequency_words_T1 = nltk.FreqDist(words_T1)
frequency_words_T1.plot(30, cumulative=False)
frequency_words_T2 = nltk.FreqDist(words_T2)
frequency_words_T2.plot(20, cumulative=False)
```



Word Cloud Snippets-

Word Cloud

In [82]:
```
def stringConverter(s):
    finstr = " "
    return (finstr.join(s))

# converting LIST OF TOKENS to a string
def tokens_to_string(words):
    text = stringConverter(words)
    return text
```

In [84]:
```
text_T1 = tokens_to_string(words_T1)
text_T2 = tokens_to_string(words_T2)
```

In [86]:
```
# word cloud for T1
wc_T1 = WordCloud(background_color="white", width=1000, height=1000, random_state=1,stopwords= [],collo
plt.imshow(wc_T1)
```

Out[86]:
```
<matplotlib.image.AxesImage at 0x185aa13e430>
```

Removing Stopword snippet-

```
In [88]: from nltk.corpus import stopwords
         nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\91916\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

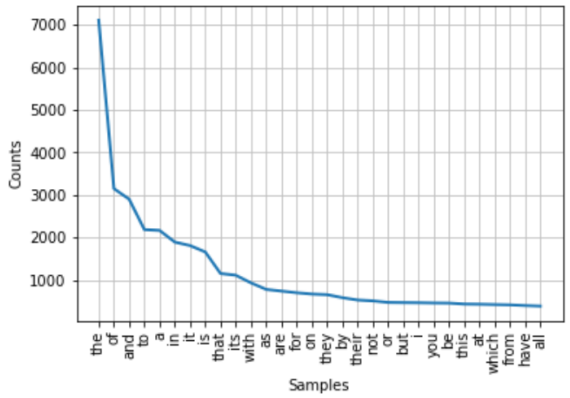
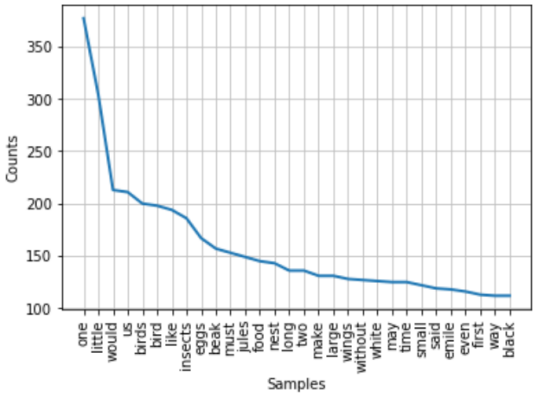
```
Out[88]: True
```

```
In [90]: # remove stopwords printed above
         def remove_stopwords(words):
             words = [w for w in words if w not in stopwords.words("english")]
             return words
```

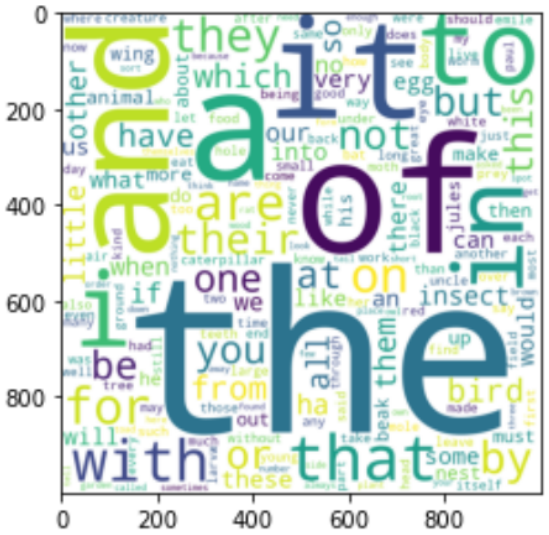
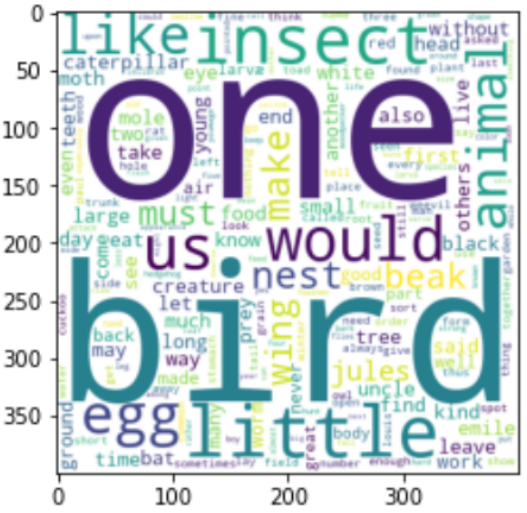
```
In [91]: nsw_words_T1 = remove_stopwords(words_T1)
         nsw_words_T2 =remove_stopwords(words_T2)
```

Illustrations -

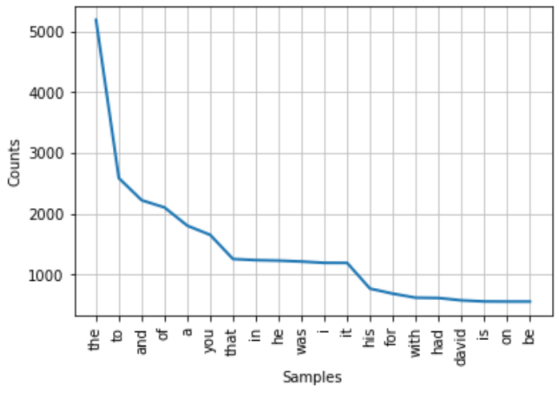
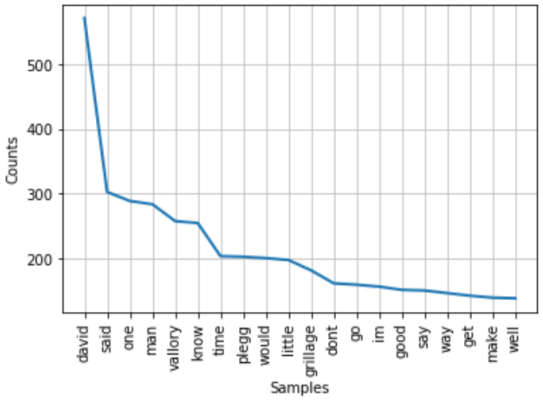
Plot of frequency analysis of T1 before and after stopword removal

Before	After
	

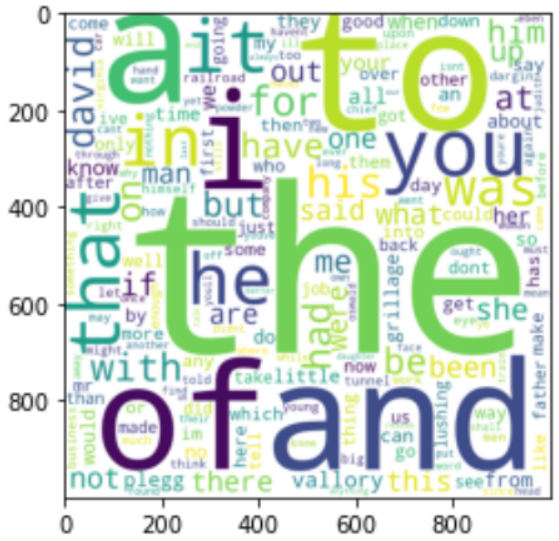
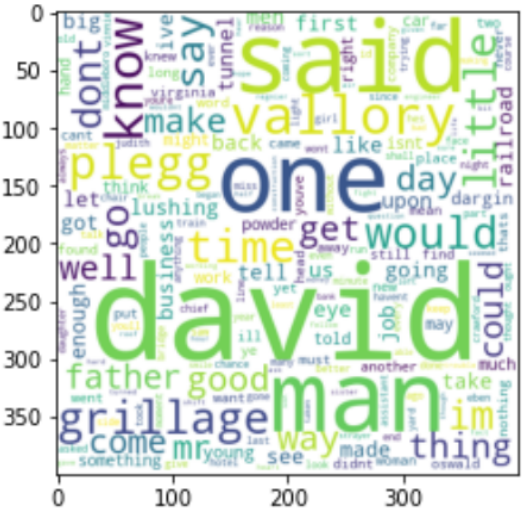
Plot of Word Cloud of T1 before and after stopword removal

Before	After
	

Plot of frequency analysis of T2 before and after stopword removal

Before	After
	

Plot of Word Cloud of T2 before and after stopword removal

Before	After
	

Inference from word Clouds and frequency plots

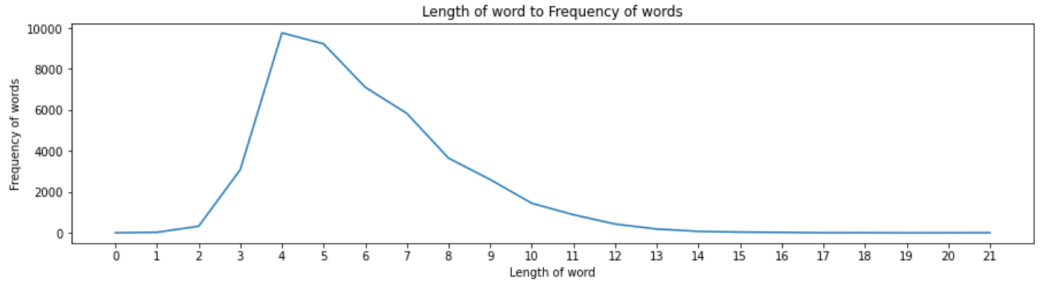
- After comparing before and after images of both frequency plot and word cloud plot for T1 and T2, we can infer Stop words have a very high frequency of occurrence. the, to, and and were the most frequent and rightly so, as these determiners are very integral to writing grammatical sentences.
- After removing stop words, we are able to find the set of words which provide us meaning and context about the documents.
- Ex-For T1(Animal life) we can see words like insects,wings, beak which clearly provide us meaning and context about the documents.

Word length to frequency plots

Illustration: Word length – frequency plots

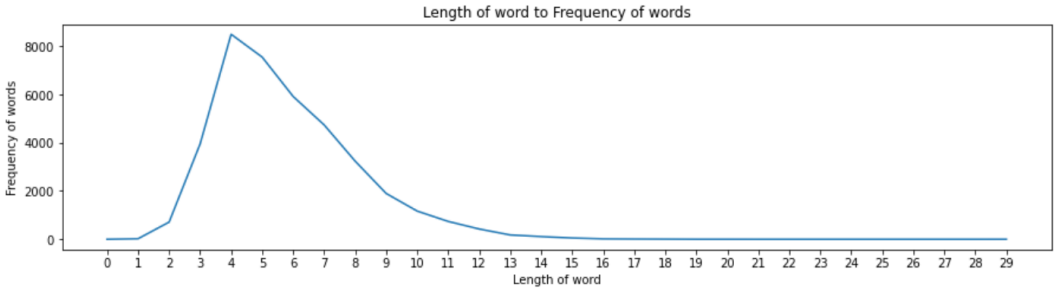
For T1-

```
In [112]: wordlength_to_frequency(nsw_words_T1)
```



For T2-

```
In [113]: wordlength_to_frequency(nsw_words_T2)
```



Inferences from word length- frequency plot

- For both the books T1 and T2 words with length between 4 to 6 have higher frequency of occurrence than other words. Very long and very small words appear very rarely. Optimal range of word length is between 4 to 6.

POS tagging -

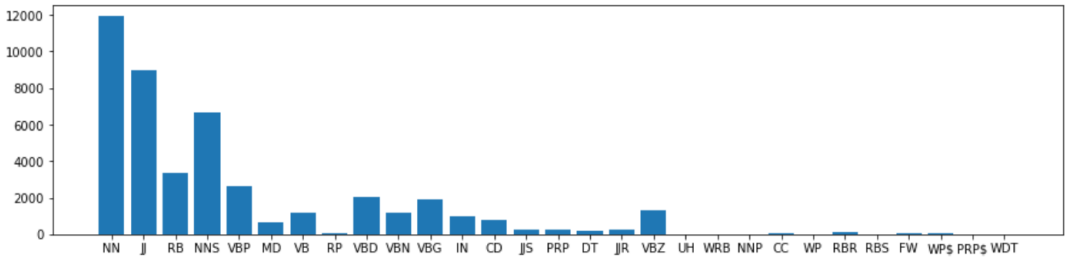
Finding the tag associated with each word that was pre-processed.

Illustrations for POS tagging

For T1

```
In [102]: dist1=get_count(res1)
          print("Number of tags used in T1=",len(dist1))
          dist1
          FrequencyPlot(dist1)
```

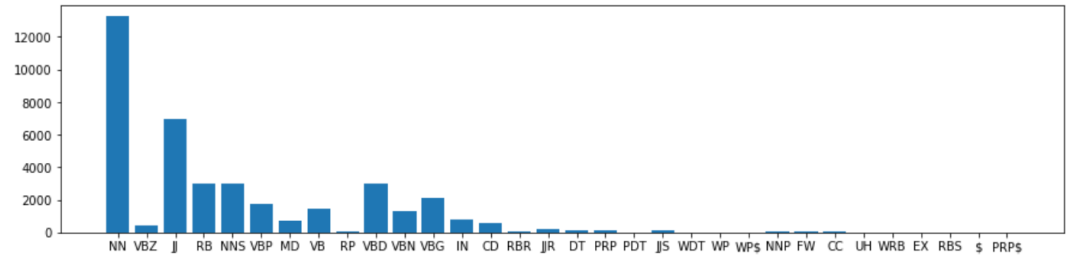
```
Number of tags used in T1= 29
```



For T2

```
In [103]: dist2=get_count(res2)
          print("Number of tags used in T2=",len(dist2))
          dist2
          FrequencyPlot(dist2)
```

```
Number of tags used in T2= 32
```



Inferences POS_tagging

Applied pos_tagging on T1 and T2.

Here are some inferences-

For T1, the most frequently occurring POS Tag is NN followed by JJ which is followed by NNS.

For T2, the most frequently occurring POS Tag is NN followed by JJ.

Conclusion

Performed some integral parts of nlp in form of tasks like word pre-processing, word tokenization, Word Cloud generation, POS tagging. Learned the requirement of preprocessing. Learned the importance of stopword removal. Learned about various pictorial representations of words, Learned the use of POS tagging.