NLP Project Round 1

Group Name - PhiGex

Group Members -

1. Ramit Agarwal 18ucs144
2. Shaswat Kumar 18ucs125
3. Shivam kejriwal 18ucs123
4. Namit Bhandari 18ucs178

Github link for the code :-
https://github.com/Ramitphi/NLP_Project

Google Colab Notebook Link :

https://colab.research.google.com/drive/12gDkWn1J8k87VtSXjfigPrCNUEyKfQ6f?usp=sharing

Index

NLP Project Round 1

1. **Data description :-**

   The data that we have taken are two text books and we have fetched the texts from it and splitted it into lines and then words.

   The code for it are given below:-

```
b1 = open("14249.txt", "r")
T1= b1.read()

b2 = open("34861.txt", "rt")
T2=b2.read()
```

2. **Data Pre-processing and preparation Steps :-**

   A) To remove the upper section containing the details of the book and removing the lower section containing the license,certification etc .

   Below is code for it :-

```
index1= T1.find("CHAPTER")#first found the index of the starting of
chapter I

index2= T1.find("End of the Project") #then we will find the index
of the starting of the lower part of the book

resT1 = T1[index1:index2]#we will use slicing to slice text between
those indexes
```

   B) The removal of redundant data and changing all the uppercase letters to lowercase for normalizing it.

   The code for it are given below:-

```
#Converting the text into lowercase.
      resT1=resT1.lower()
```

C) Remove the running chapter names by using code below:-

```
resT1 = re.sub(r'chapter \w+','',resT1)
```

D) Remove Punctuation and some special characters by using code below:-

```
resT1 = re.sub(r'[^a-zA-Z0-9\s]', '', resT1)
```

3. Problem Statements:-

3.1 Analyze the frequency distribution of tokens in T1 and T2 separately

We have done the analysis of T1 and T2 by using the following code

```
#importing the libraries
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
tokensT1df=pd.DataFrame(tokensT1)
tokensT2df=pd.DataFrame(tokensT2)
freqt1=tokensT1df.value_counts()
freqt2=tokensT2df.value_counts()
```

	Token	Counts
6793	the	3988
1	a	1735
4649	of	1606
6924	to	1566
3148	he	1410

Token Count of T1

Above code divides the text in tokens named as tokensT1 and tokensT2 and then using pandas dataframe has been created named as tokensT1df and tokensT2df and then we found the frequency of each word in both the texts.

The result that can be obtained after analysing the frequencies of both the texts is that there are some words like ‘the’ has frequency of 3998 and there are many words with frequency 1.

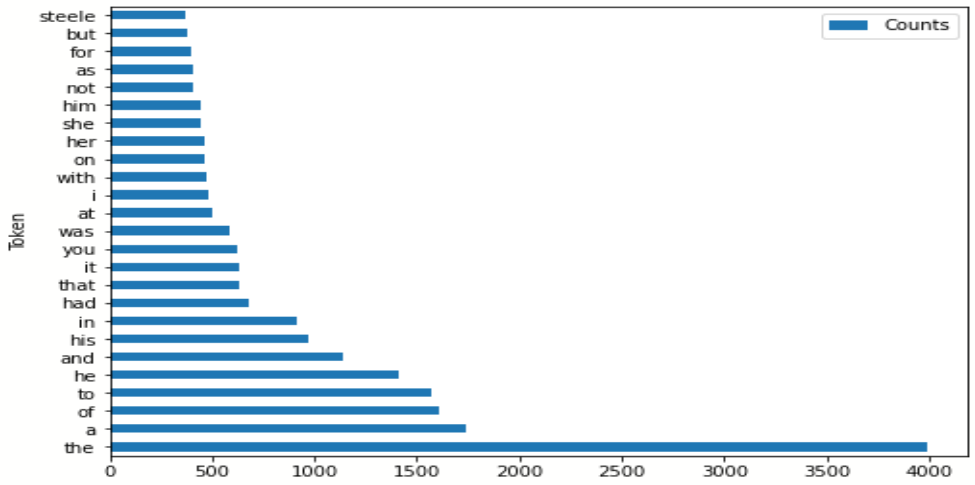
There is a huge difference in frequencies of words as we can see the words of frequency 4000 and also words of frequency 1

Considering top 25 tokens in the T1 & T2

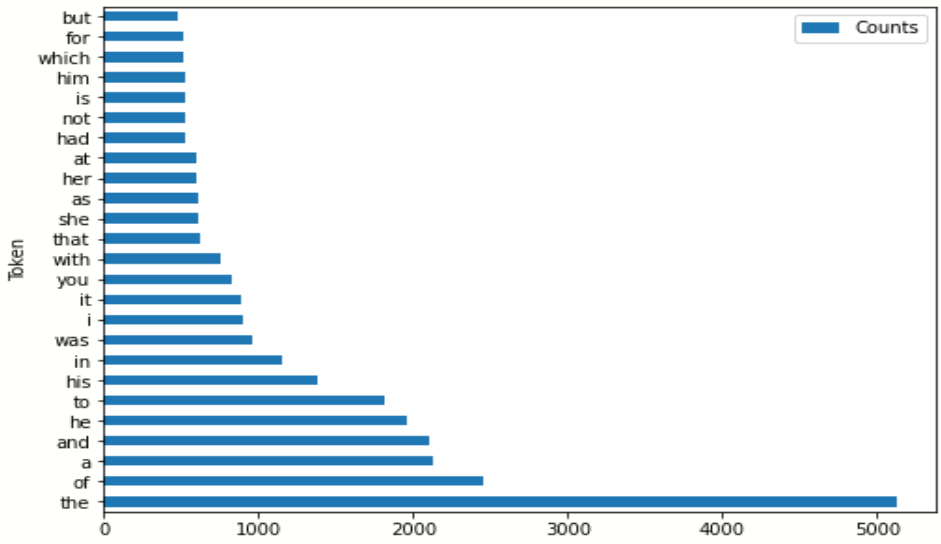
-->We find Plot of frequency for Token T1 and Token T2 by using code below-

```
t2df.plot(x= 'Token',y='Counts',kind='barh',figsize=(8, 6))
```

->Plot of Token T1 with frequency :-



B) Plot of Token T2 with frequency:-



3.2 Create a Word Cloud of T1 and T2 using the token that you have got

We have created the word cloud from the tokens we got by using the code snippet given below

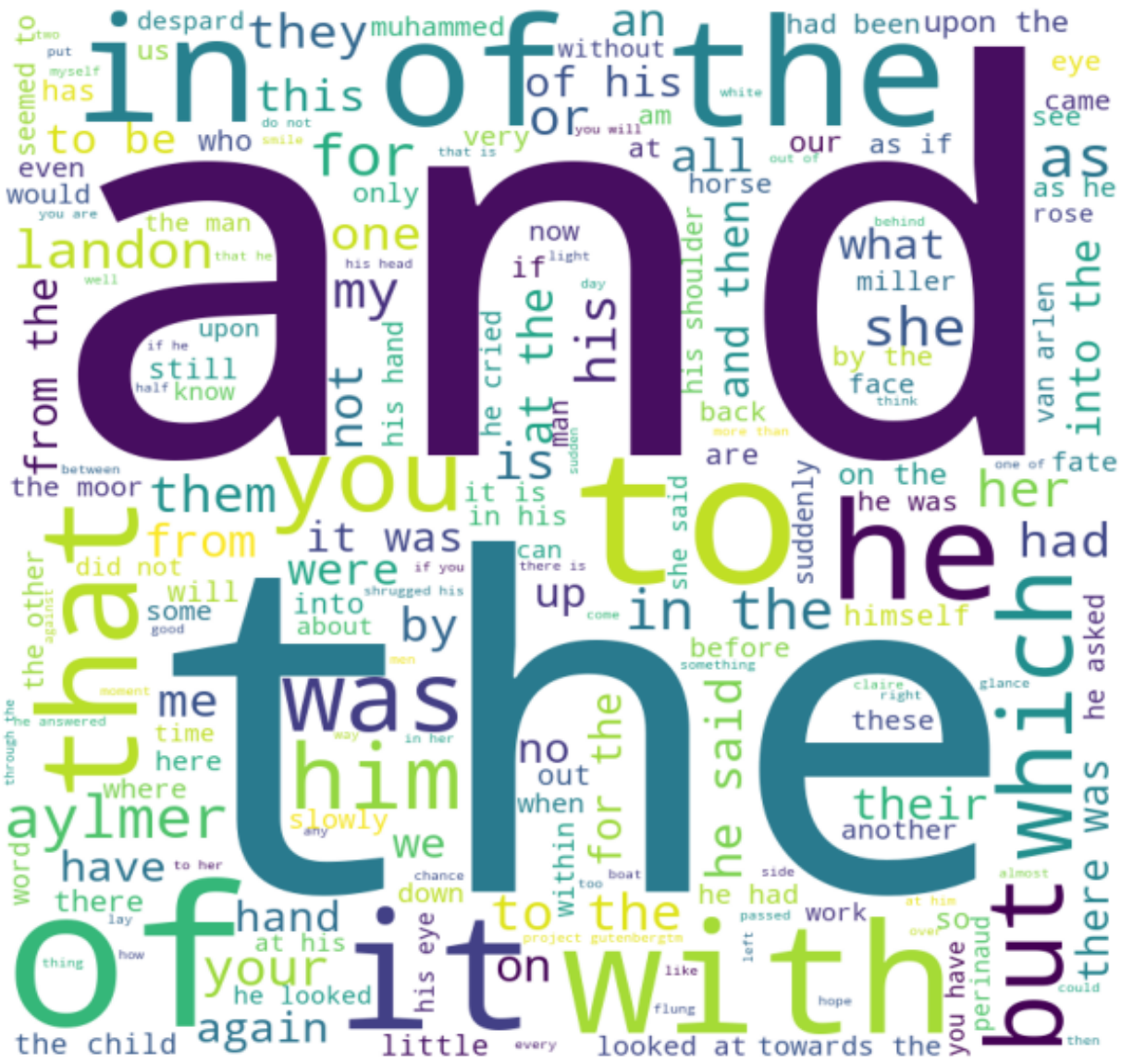
```
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd
comment_words = ''
stopwords = []
tokens = tokensT1

comment_words += " ".join(tokens)+" "
wordcloud = WordCloud(width = 800, height = 800,
        background_color ='white',
        stopwords = stopwords,
        min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
```

Using the code above we imported the word-cloud library for creating the visualization of the word cloud and then matplotlib.pyplot for plotting the words on a 2d sheet ,then we joined all the tokens in the variable named as comment words and seen in the above code snippet using .join function and then we used the inbuilt function of wordcloud and passed the parameters in it that is width and height of the sheet, background color, stopwords , and font size and after this we used the pyplot function to create the graph.

Same code is used for creating the word clouds of T2 just use the tokens of T2 in the comment_words variable.

a) WordCloud of tokenT1 with stopwords .

b) WordCloud of tokenT2 with stopwords .



3.3 Remove the stopwords from T1 and T2 and then again create a word cloud - what's the difference it gives when you compare it with word clouds before the removal of stopwords?

Removing StopWord

```
from nltk.corpus import stopwords
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
tokensT1withoutstop = [w for w in tokensT1 if not w in stop_words]
```

Similar stuff was done for the T2 to remove stopwords from it

Now from the code snippet given above are changed to remove the stopwords and the code that we use for creating word cloud without using the stopwords are:

```
comment_words = ''
tokens = tokensT1withoutstop
comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
        background_color ='white',
        min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (8, 8), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
```

In the code snippet given above the stopwords are removed for T1

The difference between those two are: the words will not be seen in the word cloud with stop words provided that is, the work of stopword is to remove the given word in the stopword from the word cloud.for eg: suppose the word ‘the’ is in the stopword then the word the will not be seen in the word cloud and the word-cloud without stopword will show the word the in it.

a) Word-cloud of TokenT1 without stop words

In a text mostly stopwords have a high frequency and after remob=ving it we can see ‘john’ , ‘steele’ and some other words are prominent in the word cloud which shows that these words were widely used in the book apart from the stopwords

b) Word-cloud of TokenT2 without stop words

In a text mostly stopwords have a high frequency and after remob=ving it we can see ‘aylmer’ , ‘landon’ and some other words are prominent in the word cloud which shows that these words were widely used in the book apart from the stopwords

---

3.4 Evaluate the relationship between the word length and frequency for both T1 and T2- what's your result?

The relationship between world length and frequency of both T1 and T2 can be given by the following code snippet below

```
tokensT1df=pd.DataFrame(tokensT1)
tokensT2df=pd.DataFrame(tokensT2)
freqt1=tokensT1df.value_counts()
freqt1= pd.DataFrame(freqt1,columns=['freq'])
freqt1w= freqt1.reset_index()
freqt1w.head()
```

	tokens	frequency
0	the	3988
1	a	1735
2	of	1606
3	to	1566
4	he	1410

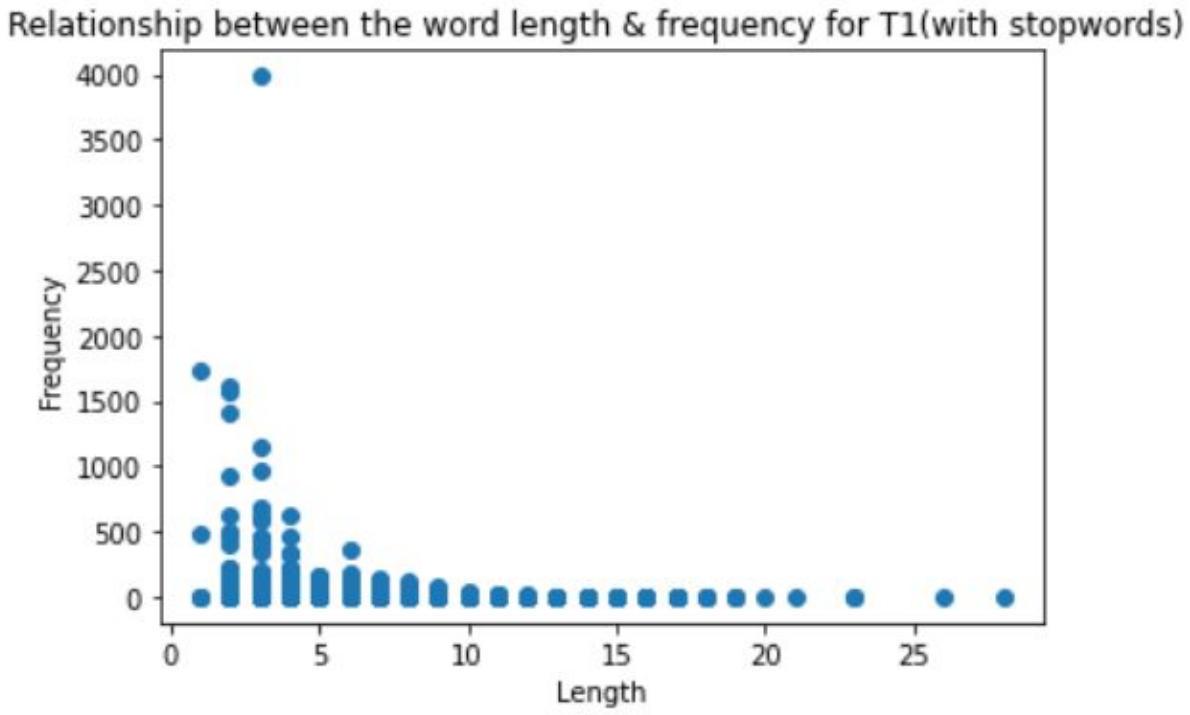
```
freqt1w['length'] = freqt1w['tokens'].apply(lambda x: len(x))
#using the apply method to apply the len() function to all values of a
column
```

	tokens	frequency	length
0	the	3988	3
1	a	1735	1
2	of	1606	2
3	to	1566	2
4	he	1410	2

```
plt.scatter(freqt1w['length'], freqt1w['frequency'])

plt.title("Relationship between the word length & frequency for T1(with
stopwords) ")
plt.ylabel("Frequency")
plt.xlabel("Length")
```

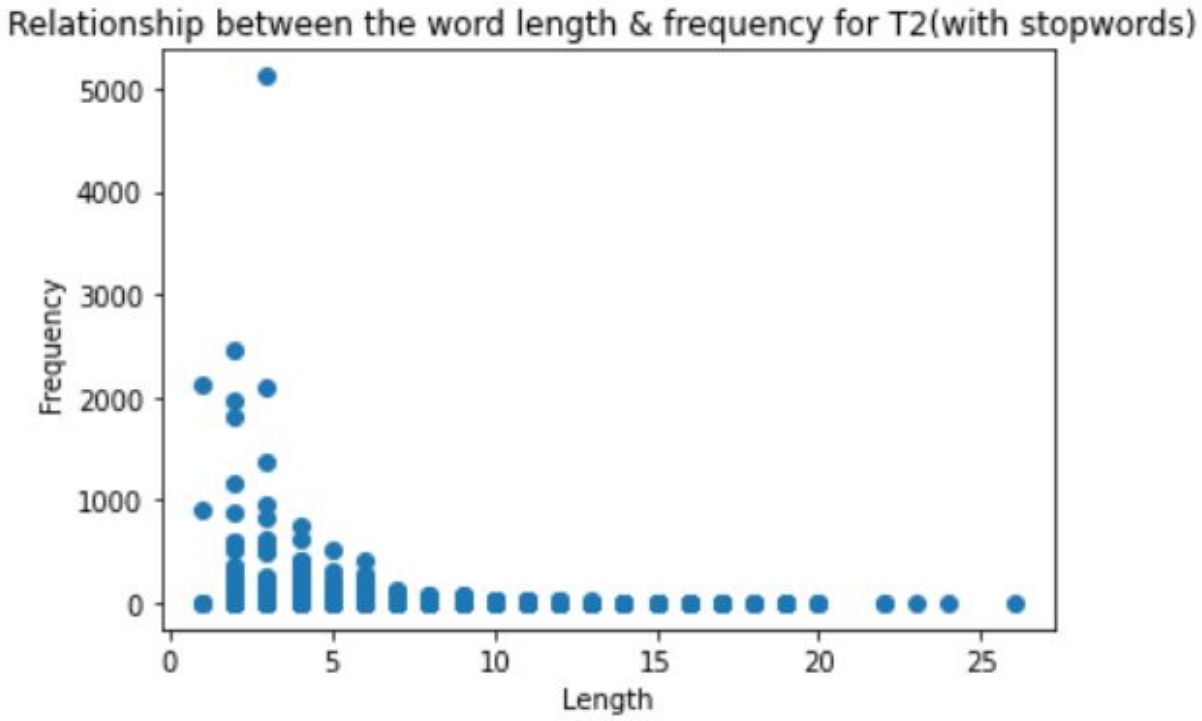
Above code snippet will have counts of all the words that will be stored in wordlen_count_list and each word-length will be plotted against it's frequency . That is shown below in the graph:-

T1 graph->



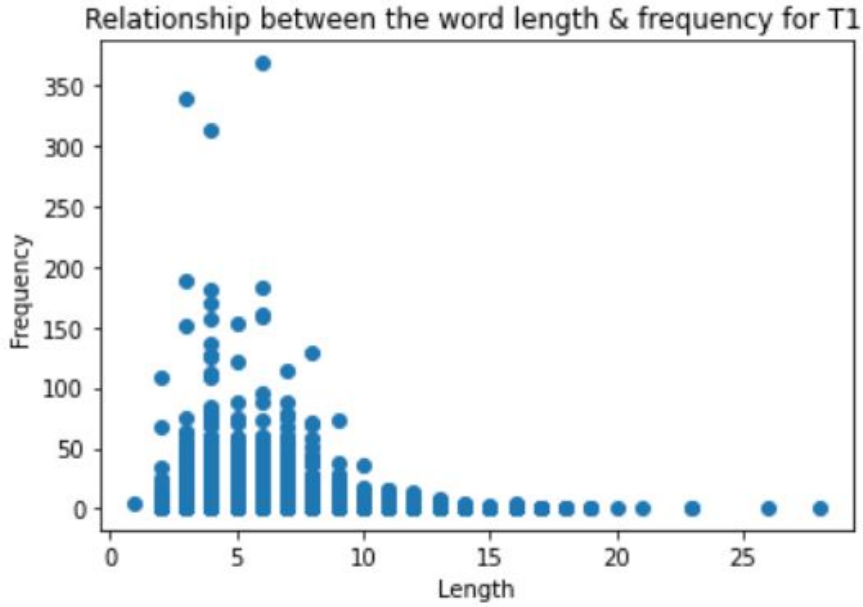
Word with world length from 1 to 5 have high frequency as compared to longer words

T2 graph->



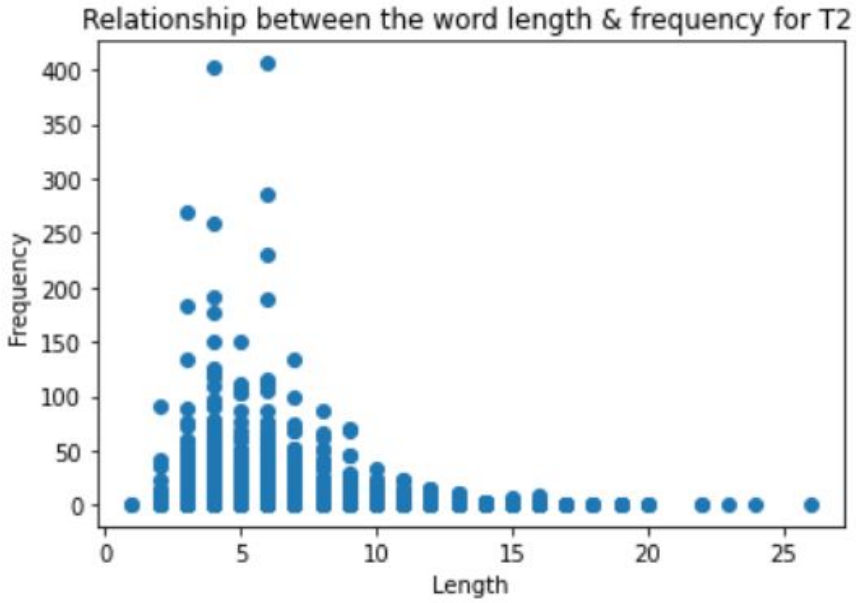
Similar plot for the T1 & T2 without Stopwords

T1(without stopwords) >>



Word from length 1 to 10 are having good number of occurrences

T2(without stopwords) >>



3.5 Do PoS Tagging for both T1 and T2 using anyone of the four tagset studied in the class and Get the distribution of various tags

We have assigned tag to every token by using function of Pos Tagging in NLTK .The Code Snippet is given below :-

```
import nltk;
nltk.download('averaged_perceptron_tagger');
tags_T1 = nltk.pos_tag(tokensT1)
tag1 = [ t for (w,t) in tags_T1]
t1tags, t1tagcount = np.unique(tag1, return_counts=True)
t1tagcountdf = pd.DataFrame(list(zip(t1tags, t1tagcount)),
          columns =['Tags', 'Counts']);
t1tagcountdf.sort_values(by=['Counts'],ascending=False,inplace=True)
t1toptagdf= t1tagcountdf[:30]
t1toptagdf.plot(x= 'Tags',y='Counts',kind='barh',figsize=(8, 6))
```

```
t1tagcountdf.head()
```

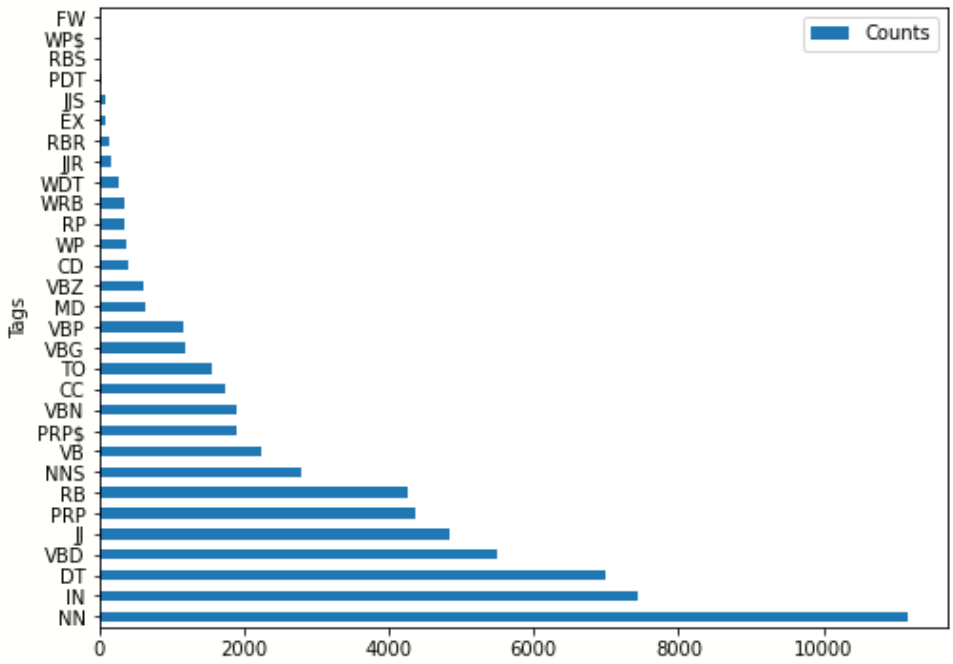
	Tags	Counts
0	CC	1750
1	CD	408
2	DT	6987
3	EX	98
4	FW	17

```
t2tagcountdf.head()
```

	Tags	Counts
0	CC	2811
1	CD	520
2	DT	8847
3	EX	240
4	FW	18

We have calculated count of each tag associated with tokens and plot of tag and frequency is given below:

-> **Plot for token T1:**



Inference : Noun occur predominantly in the text followed by preposition and by determiners

-> Plot for Token T2: