NLP Project Round-1

Github Link: https://github.com/mayank-gupta16/Latent-Dirichlet

Team Name: Latent Dirichlet

Members:

Saransh Mittal(20UCS174)

Aagam Jain(20UCS001)

Mayank Gupta(20UCS113)

Mihir Chaturvedi(20UCS117)

Task:

1. **Import the text, let's call it as T1**
2. **Perform simple text pre-processing steps**
3. **Tokenise the text T1**
4. **Analyse the frequency distribution of tokens in T1.**
5. **Create a Word Cloud of T1**
6. **Remove the stop words from T1 and then again create a word cloud**
7. **Evaluate the relationship between the word length and frequency for T1**
8. **Do PoS Tagging for T1 using anyone of the four tag sets studied in the class and get the distribution of various tags**

Library Used

NLTK - Used for tokenizing, lemmatization and removing stopwords.

Language Word Cloud - Used to create word clouds from tokenized data.

Matplotlib- Used to Visualize our text data

Seaborn -Used to Visualize our text data

Numpy- Used for working with arrays.

Typing-To perform evaluation of the Algorithm in Entity Recognition

IMPORT THE TEXT T1

```
# Load data
file = open('T1.txt', encoding='utf-8')
text = file.read()
file.close()
```

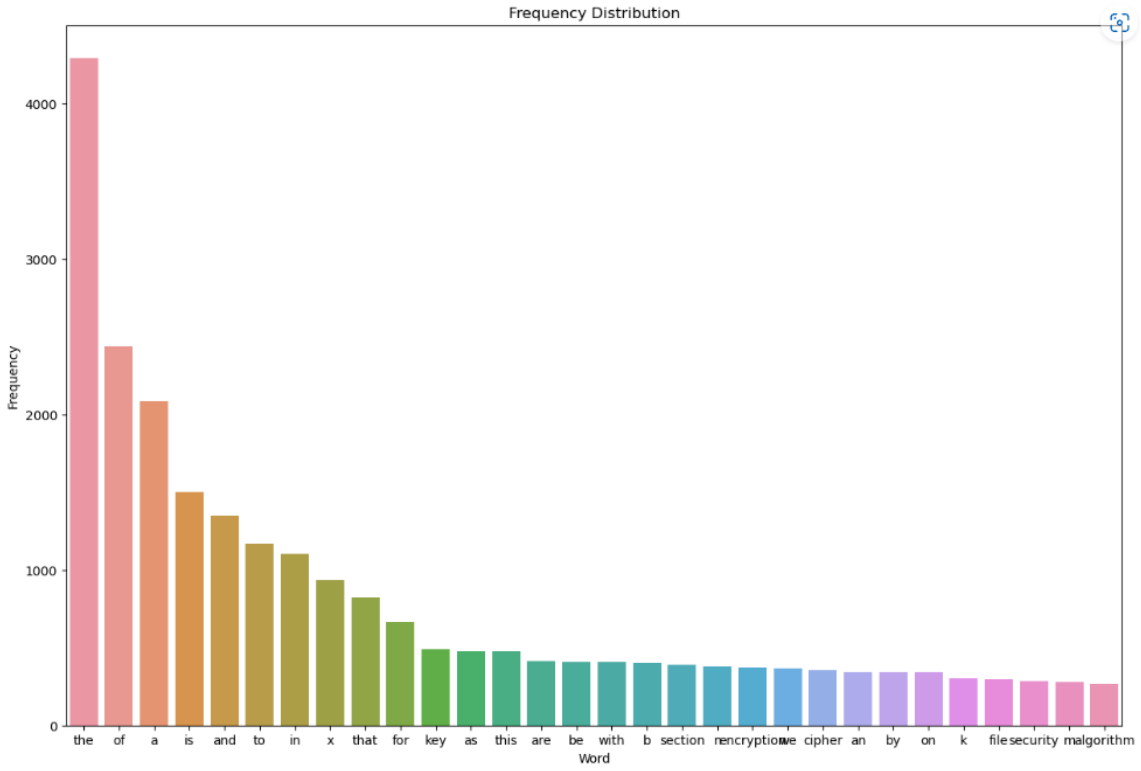
TEXT PREPROCESSING STEPS

- Here we have used word_tokenize of nltk library for splitting the text into words.

- Then we have converted the upper case tokens to lower case tokens .
- Then we removed the punctuations and digits.
- Then we also have removed the remaining tokens which are not alphabetical.
- Then we exclude stop words for which we have used nltk.corpus

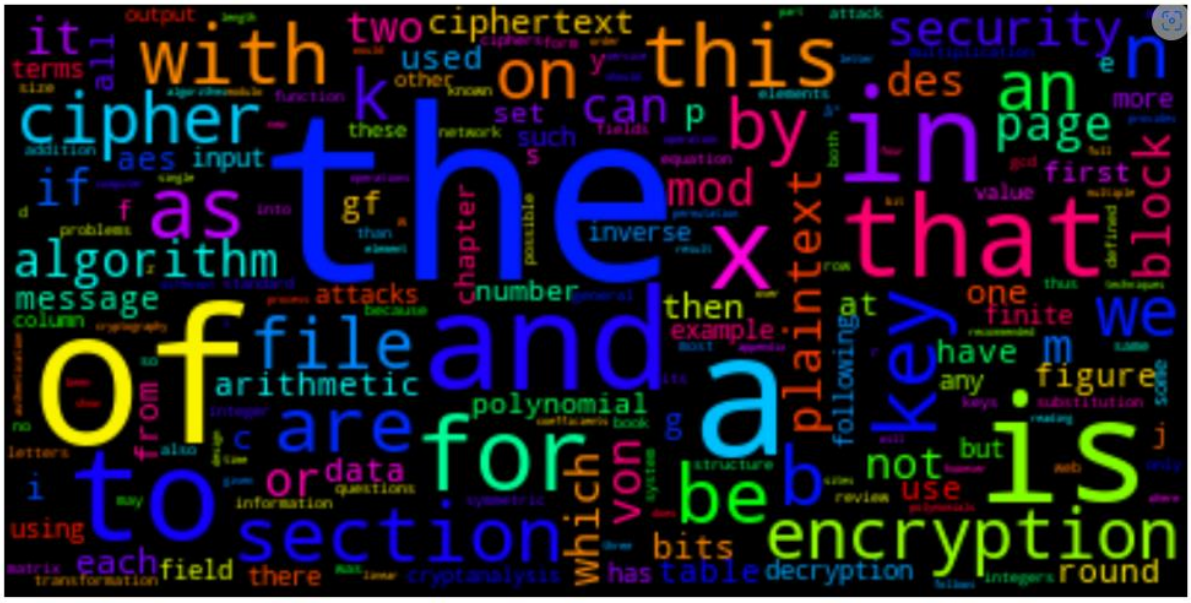
Frequency Distribution Including stopwords

```
# frequency distribution of tokens in T1 including stopwords
import string
from nltk import FreqDist
from nltk.corpus import stopwords
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import nltk
#nltk.download('all')

# load data
file = open('T1.txt', encoding='utf-8')
text = file.read()
file.close()

# split into words
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set( list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
filtered_text = [word for word in filtered_text if word.isalpha()]
## Creating FreqDist  keeping the 30 most common tokens
all_fdist = FreqDist(filtered_text).most_common(30)
all_fdist = pd.Series(dict(all_fdist))
fig, ax = plt.subplots(figsize=(15,10))
plt.title('Frequency Distribution')
plt.ylabel('Frequency')
plt.xlabel('Word')
all_plot = sns.barplot(x=all_fdist.index, y=all_fdist.values, ax=ax)
```



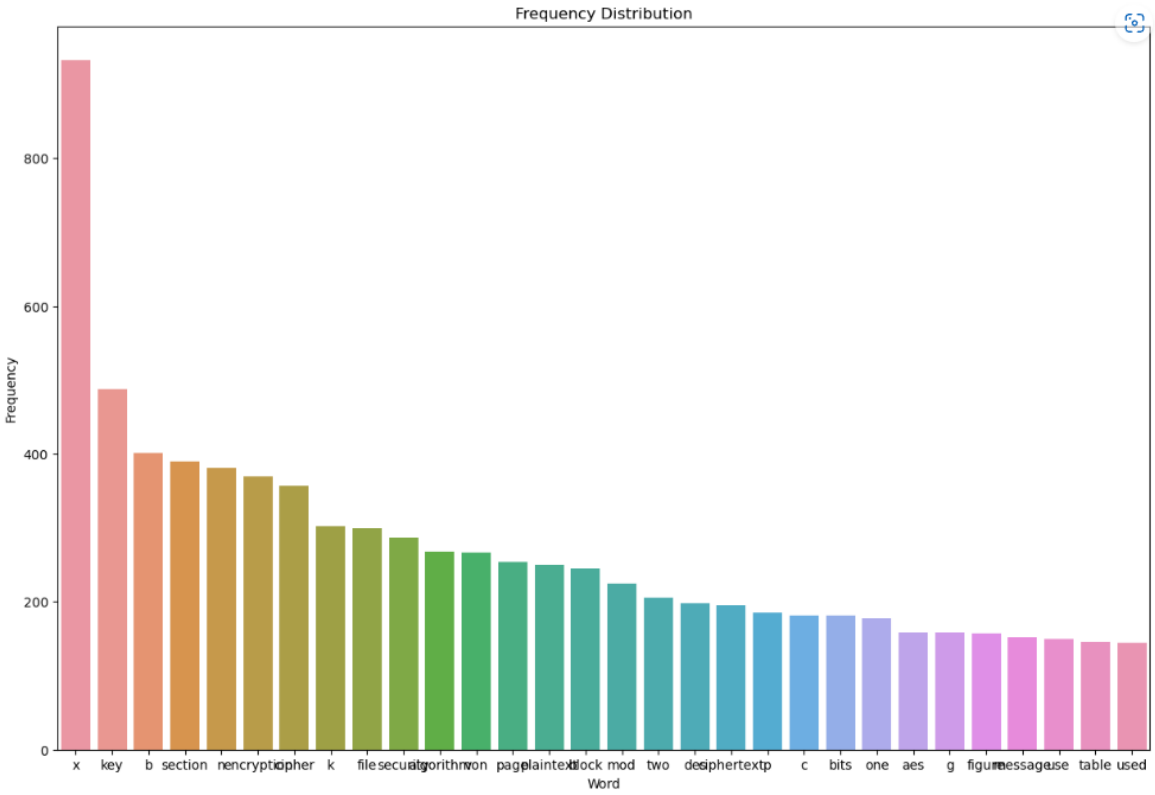
Word Cloud Including stopwords

```
import string
from nltk import FreqDist
from nltk.corpus import stopwords

# split into words
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set(list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
filtered_text = [word for word in filtered_text if word.isalpha()]

from collections import Counter
dictionary=Counter(filtered_text)
import matplotlib.pyplot as plt
from wordcloud import WordCloud

cloud = WordCloud(max_font_size=80,colormap="hsv").generate_from_frequencies(dictionary)
plt.figure(figsize=(16,12))
plt.imshow(cloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



Frequency Distribution excluding stopwords

```
import string
from nltk import FreqDist
from nltk.corpus import stopwords
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set(stopwords.words('english') + list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
filtered_text = [word for word in filtered_text if word.isalpha()]
## Creating FreqDist  keeping the 30 most common tokens
all_fdist = FreqDist(filtered_text).most_common(30)
all_fdist = pd.Series(dict(all_fdist))
fig, ax = plt.subplots(figsize=(15,10))
plt.title('Frequency Distribution')
plt.ylabel('Frequency')
plt.xlabel('Word')
all_plot = sns.barplot(x=all_fdist.index, y=all_fdist.values, ax=ax)
```



Word Cloud Exluding stopwords

```
import string
from nltk import FreqDist
from nltk.corpus import stopwords

# split into words
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set(stopwords.words('english') + list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
filtered_text = [word for word in filtered_text if word.isalpha()]

from collections import Counter
dictionary=Counter(filtered_text)
import matplotlib.pyplot as plt
from wordcloud import WordCloud

cloud = WordCloud(max_font_size=80,colormap="hsv").generate_from_frequencies(dictionary)
plt.figure(figsize=(16,12))
plt.imshow(cloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

Inferences:

- Words like ‘of’, ‘a’ and ‘the’ are the most frequently used words in T1
- Words like ‘of’, ‘and’, ‘a’, and ‘the’ are frequently used words in T2
- These words do not contribute to the meaning of the sentence and are mostly useless for us

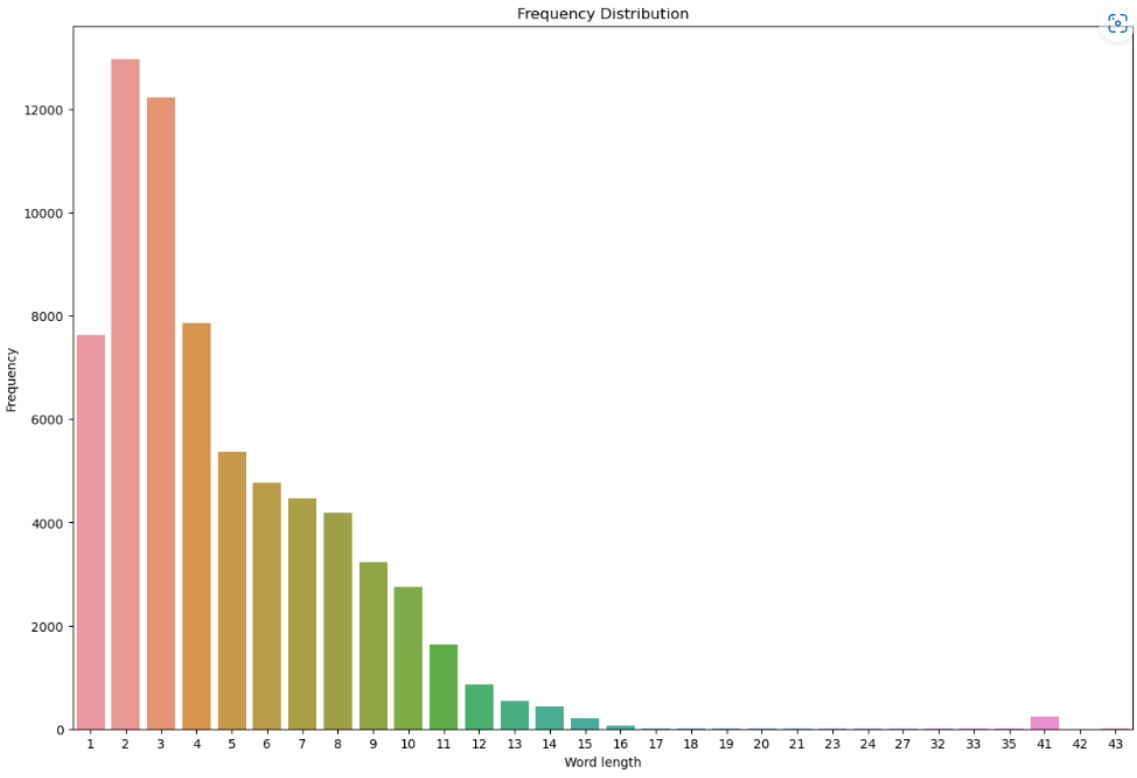
These words are known as ‘stopwords’, and we need to get rid of them

On excluding stop words, we get a meaningful word cloud as stop words mostly have higher frequency due to which our result might get affected. Now most of the terms that are of no meaning to us have been removed. We have gotten rid of ‘stopwords’ and can draw meaningful conclusions from the data.

Relation between word length and frequency for Text

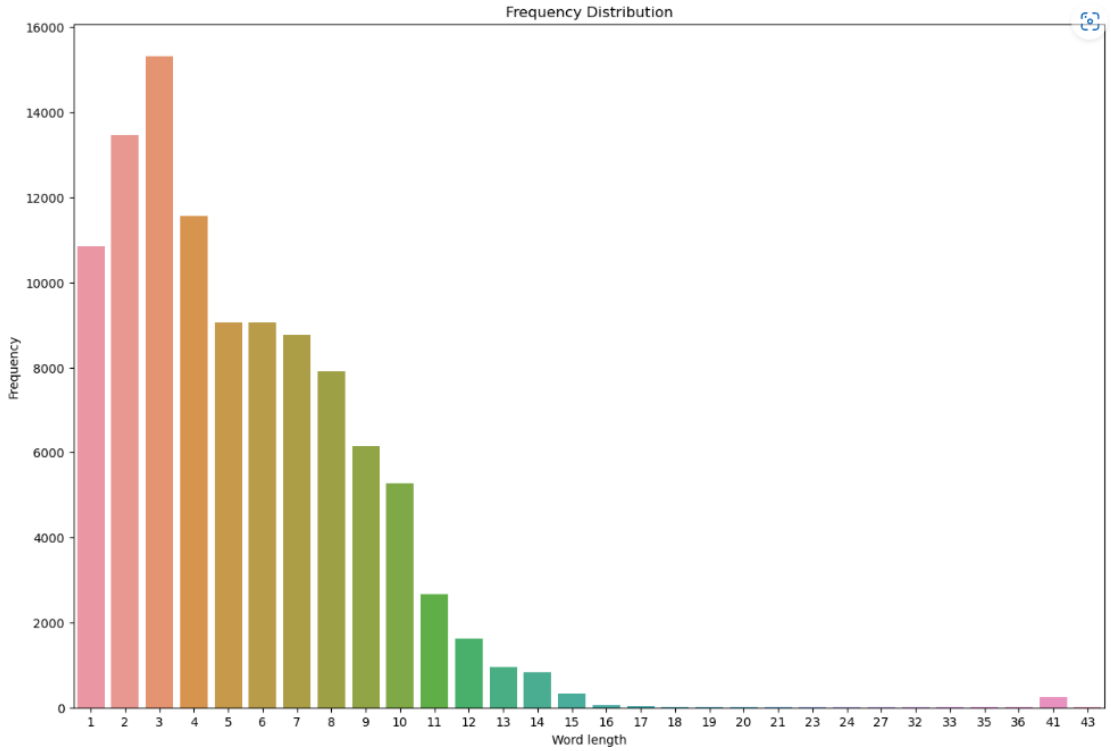
Including Stopwords:

```python
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

words = text.split()
lengths = {}
for word in words:
    length = len(word)
    if length not in lengths:
        lengths[length] = 1
    else:
        lengths[length] += 1
## Creating FreqDist  keeping the 30 most common tokens
all_fdist = FreqDist(lengths).most_common(30)
all_fdist = pd.Series(dict(all_fdist))
fig, ax = plt.subplots(figsize=(15,10))
plt.title('Frequency Distribution')
plt.ylabel('Frequency')
plt.xlabel('Word length')
all_plot = sns.barplot(x=all_fdist.index, y=all_fdist.values, ax=ax)
```



Excluding StopWords:

```
import string
from nltk import FreqDist
from nltk.corpus import stopwords
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

# split into words
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set(stopwords.words('english') + list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
words = [word for word in filtered_text if word.isalpha()]

for word in words:
    length = len(word)
    if length not in lengths:
        lengths[length] = 1
    else:
        lengths[length] += 1
## Creating FreqDist  keeping the 30 most common tokens
all_fdist = FreqDist(lengths).most_common(30)
all_fdist = pd.Series(dict(all_fdist))
fig, ax = plt.subplots(figsize=(15,10))
plt.title('Frequency Distribution')
plt.ylabel('Frequency')
plt.xlabel('Word length')
all_plot = sns.barplot(x=all_fdist.index, y=all_fdist.values, ax=ax)
```



Inferences:

- The number of words of length 1 has significantly increased after removing stopwords as now it will also count varialbles like ‘x’ in it.
- There is a general trend that the highest number of words lies in the length range 3-6.
- We can infer that this is due to removing stop words like ‘a’, ‘the’, ‘of’ and ‘and’, which were the highest occurring words before removal.

- **POS TAGGING**
- Now the main task is to give each of the tokens their tags. We have used
- ‘averaged_perceptron_tagger’ for POS tagging of the tickets. We have downloaded the same from nltk.

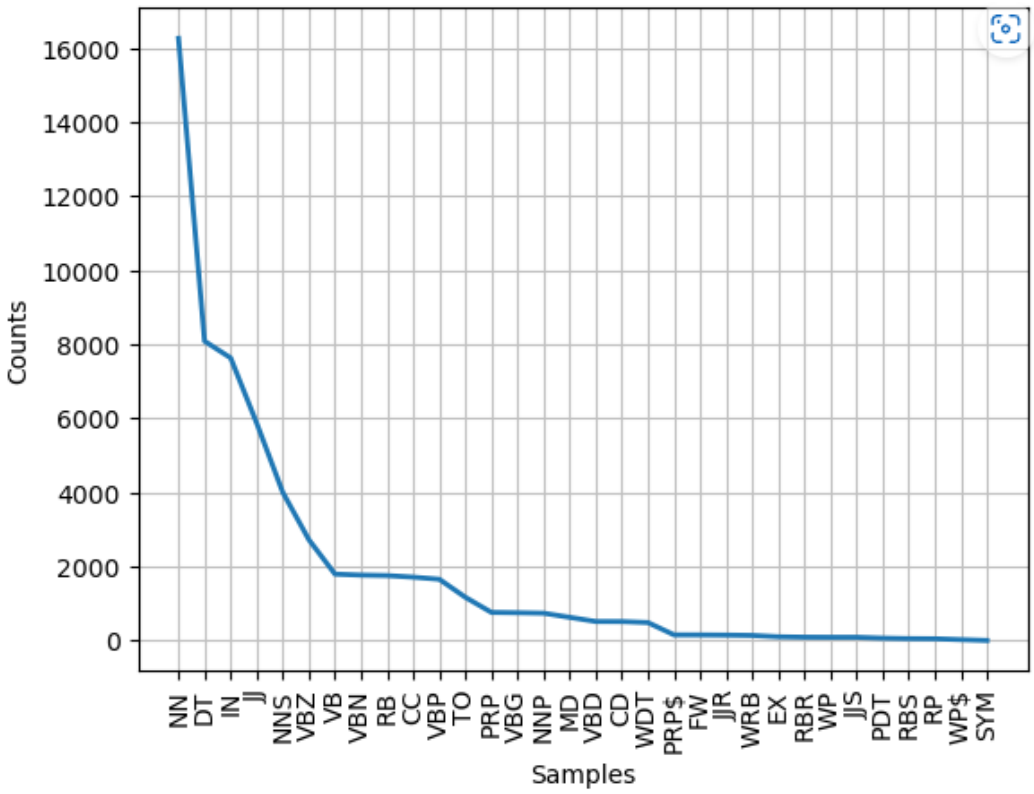
```
from nltk.corpus import stopwords

# split into words
from nltk.tokenize import word_tokenize
tokens = word_tokenize(text)
# convert to lower case
tokens = [w.lower() for w in tokens]
remove_these = set( list(string.punctuation) + list(string.digits))
filtered_text = [w for w in tokens if not w in remove_these]
# remove remaining tokens that are not alphabetic
token1 = [word for word in filtered_text if word.isalpha()]

tagged1=nltk.pos_tag(token1)
tagged1
```

```
[('cryptography', 'NN'),
 ('and', 'CC'),
 ('network', 'NN'),
 ('security', 'NN'),
 ('principles', 'NNS'),
 ('and', 'CC'),
 ('practices', 'NNS'),
 ('fourth', 'JJ'),
 ('edition', 'NN'),
 ('cryptography', 'NN'),
 ('and', 'CC'),
 ('network', 'NN'),
 ('security', 'NN'),
 ('principles', 'NNS'),
 ('and', 'CC'),
 ('practices', 'NNS'),
 ('fourth', 'JJ'),
 ('edition', 'NN'),
 ('by', 'IN'),
 ('william', 'JJ'),
 ('stallings', 'NNS'),
 ('publisher', 'NN'),
 ('prentice', 'NN'),
 ('hall', 'NN'),
 ('pub', 'NN'),
 ('date', 'NN'),
 ('november', 'JJ'),
 ('print', 'NN'),
 ('print', 'NN'),
 ('etext', 'RBR'),
 ('table', 'NN'),
 ('of', 'IN'),
 ('contents', 'NNS'),
 ('etext', 'JJ'),
```

Frequency Distribution of Tags:

```
def FrequencyDist(tags):
    wfd=FreqDist(t for (w, t) in tags)
    wfd
    wfd.plot(50)
```

```
FrequencyDist(tagged1)
```



Inferences

From the above results we infer that the highest occurring tag is 'NN', and 'Determinant' Tags are on the lower frequency side. This is largely due to the removal of stopwords before POS Tagging.

Conclusion:

We have learnt how to perform Text Preprocessing, Tokenization on Text Data and how to Create word clouds. We have solved all the Problems given to us in NLP Project Round 1 with the use of Plots and Visualisations and learnt to draw meaningful inferences from it.