Harshit Mishra (19BCE0799)

CSE-3024 WEB MINING

LAB ASSIGNMENT 2

Aim: Tokenize an input text file using NLTK and remove the stop words from the tokens.

Procedure:

- Firstly, we import the text file in our work space. To do this we can use open method of python which reads the file into our workspace.
- Next, we import the necessary NLTK libraries including stopwords, sent_tokenize and word_tokenize.
- Next, using regex in python we remove the punctuations from our string input. This will make sure that tokens are free from sentence structure.
- Using word_tokenize we tokenize each word and store it in a variable list named tokens
- Then using Numpy's unique method we include only unique tokens from our text.
- To remove stopwords, we use NLTK's stopwords library and a nested loop that ensures all the words not present in that library are added onto the result list.
- Finally, we print our list that contains the resultant tokens post removal of stop words as well.

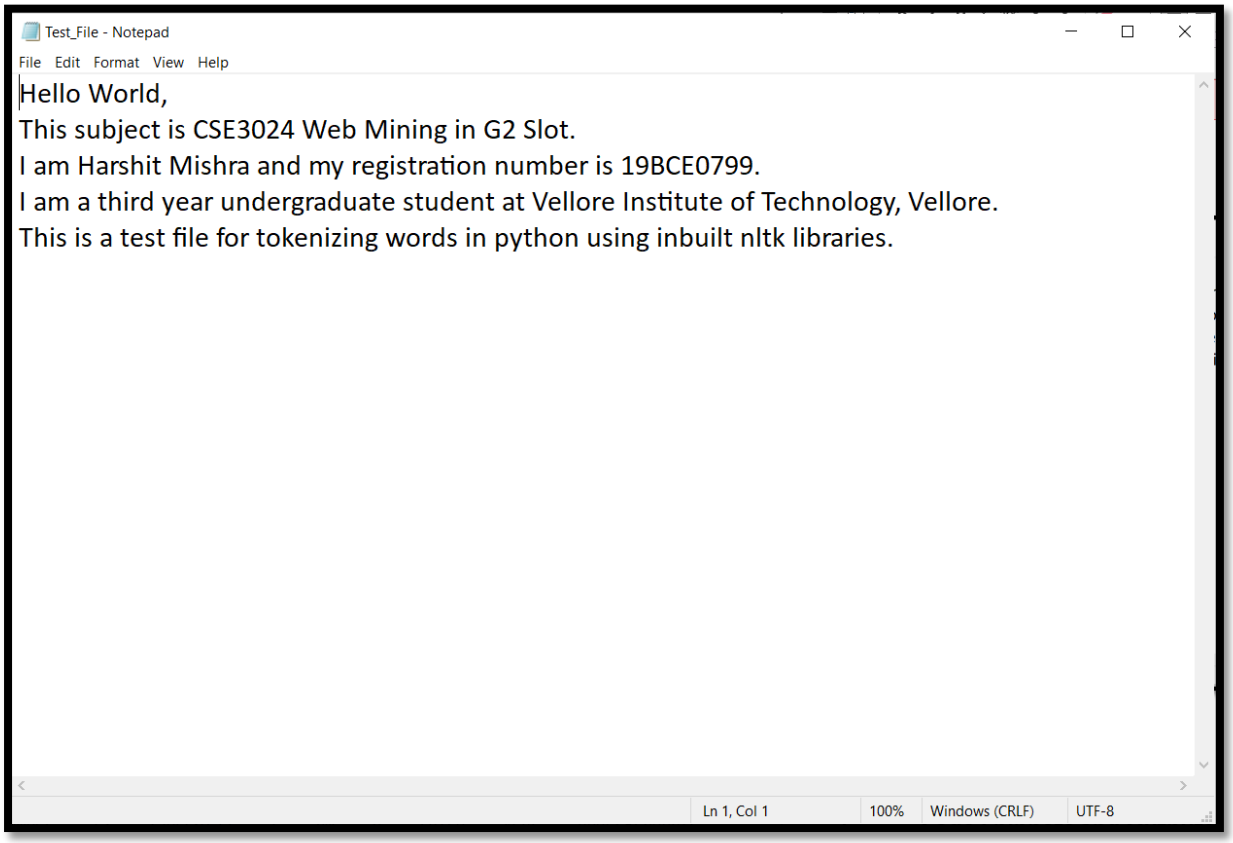
Harshit Mishra (19BCE0799)

Code:

```
#Reading input from a text file and saving it as a string
text = ""
with open('test_file.txt') as file:
   for line in file:
      for word in line.split():
         text= text + " " + word

#Importing libraries
import re
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

#Removing punctuations from our input file
import re
text = re.sub(r'[^\w\s]', '', text)
text

#Printing each token
tokens = word_tokenize(text)
print(tokens)

#Printing unique tokens
import numpy as np
tokens = np.unique(tokens)
print(tokens)

#Removing Stopwords
res = []
for x in tokens:
   if x not in set(stopwords.words('english')):
      res.append(x)
print(res)


#Printing cleaned tokens in our input text file
print(res)
```

Harshit Mishra (19BCE0799)

Text File Taken as Input:



Code Snippets and Outputs:

```
In [1]: #Reading input from a text file and saving it as a string
        text = ""
        with open('test_file.txt') as file:
            for line in file:
                for word in line.split():
                    text= text + " " + word
```

Here we are reading the text file using open method in python. Then reading each line we split each word and append it to a string variable with a space in between.

```
In [2]: #Importing libraries
        import re
        import nltk
        from nltk.tokenize import sent_tokenize, word_tokenize
        from nltk.corpus import stopwords
```

Here we are importing the necessary libraries which includes our library of concern that is nltk. We also import stopwords and word_tokenize, sentence_tokenize from nltk. To remove punctuations we import the regex library.

```
In [3]: #Removing punctuations from our input file
        text = re.sub(r'[^\w\s]', '', text)
        text

Out[3]: ' hello world this subject is cse3024 web mining in g2 slot i am harshit mish
        ra and my registration number is 19bce0799 i am a third year undergraduate st
        udent at vellore institute of technology vellore this is a test file for toke
        nizing words in python using inbuilt nltk libraries'
```

Here we are removing punctuations from our input file. This is done using regex, where we keep only alphanumeric inputs in our text string. We can see all the periods and commas from original input files are removed here.

```
In [4]: #Printing each token
        tokens = word_tokenize(text)
        print(tokens)

        ['hello', 'world', 'this', 'subject', 'is', 'cse3024', 'web', 'mining', 'in',
        'g2', 'slot', 'i', 'am', 'harshit', 'mishra', 'and', 'my', 'registration', 'n
        umber', 'is', '19bce0799', 'i', 'am', 'a', 'third', 'year', 'undergraduate',
        'student', 'at', 'vellore', 'institute', 'of', 'technology', 'vellore', 'thi
        s', 'is', 'a', 'test', 'file', 'for', 'tokenizing', 'words', 'in', 'python',
        'using', 'inbuilt', 'nltk', 'libraries']
```

Since we have to tokenize each work, we have used the word_tokenize and as we can see each token in identified here and we have printed them.

```
In [5]: #Printing unique tokens
        import numpy as np
        tokens = np.unique(tokens)
        print(tokens)

        ['19bce0799' 'a' 'am' 'and' 'at' 'cse3024' 'file' 'for' 'g2' 'harshit'
         'hello' 'i' 'in' 'inbuilt' 'institute' 'is' 'libraries' 'mining' 'mishra'
         'my' 'nltk' 'number' 'of' 'python' 'registration' 'slot' 'student'
         'subject' 'technology' 'test' 'third' 'this' 'tokenizing' 'undergraduate'
         'using' 'vellore' 'web' 'words' 'world' 'year']
```

Here we are only printing unique tokens from all the generated tokens using split method. This is done using NumPy's unique function, which identifies all the unique elements from a list.

```
In [6]: #Removing Stopwords
        res = []
        for x in tokens:
            if x not in set(stopwords.words('english')):
                res.append(x)

In [7]: #Printing cleaned tokens in our input text file
        print(res)

        ['19bce0799', 'cse3024', 'file', 'g2', 'harshit', 'hello', 'inbuilt', 'instit
        ute', 'libraries', 'mining', 'mishra', 'nltk', 'number', 'python', 'registrat
        ion', 'slot', 'student', 'subject', 'technology', 'test', 'third', 'tokenizin
        g', 'undergraduate', 'using', 'vellore', 'web', 'words', 'world', 'year']
```

Here we remove all the stop words from a self-defined list of stop words. We use nested loop to check if given token belongs to both tokens list and stopwords list. If it does, we don't add it to our result else we add it to our results.

Results and Output:

- Input text:

```
Hello World,
This subject is CSE3024 Web Mining in G2 Slot.
I am Harshit Mishra and my registration number is 19BCE0799.
I am a third year undergraduate student at Vellore Institute of Technology, Vellore.
This is a test file for tokenizing words in python using inbuilt nltk libraries.
```

- Tokens of input text:

```
['hello', 'world', 'this', 'subject', 'is', 'cse3024', 'web', 'mining', 'in',
'g2', 'slot', 'i', 'am', 'harshit', 'mishra', 'and', 'my', 'registration', 'n
umber', 'is', '19bce0799', 'i', 'am', 'a', 'third', 'year', 'undergraduate',
'student', 'at', 'vellore', 'institute', 'of', 'technology', 'vellore', 'thi
s', 'is', 'a', 'test', 'file', 'for', 'tokenizing', 'words', 'in', 'python',
'using', 'inbuilt', 'nltk', 'libraries']
```

- Tokens after removal of stop words:

```
['19bce0799', 'cse3024', 'file', 'g2', 'harshit', 'hello', 'inbuilt', 'instit
ute', 'libraries', 'mining', 'mishra', 'nltk', 'number', 'python', 'registrat
ion', 'slot', 'student', 'subject', 'technology', 'test', 'third', 'tokenizir
g', 'undergraduate', 'using', 'vellore', 'web', 'words', 'world', 'year']
```

Harshit Mishra (19BCE0799)

Question

a) Aim: Tokenize a single sentence text using NLTK.

b) Aim: Tokenize a multi-line text using NLTK.

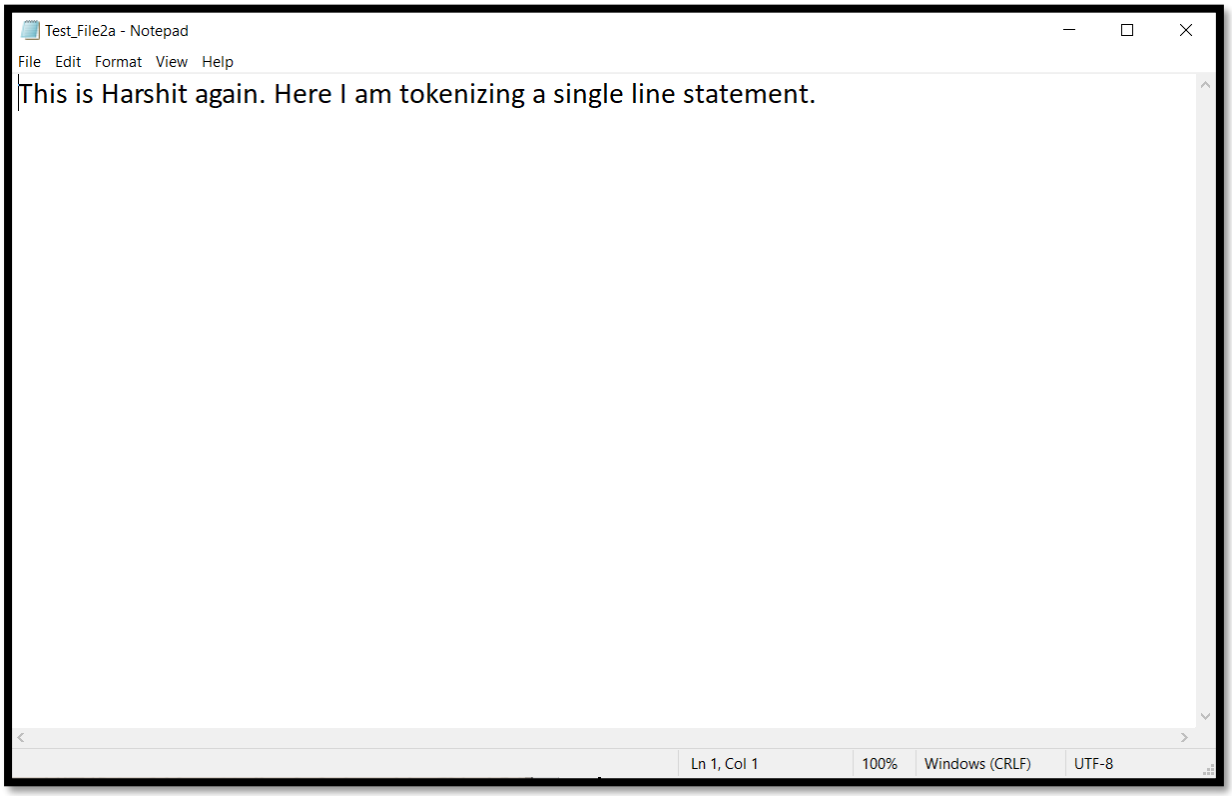
Procedure:

- Firstly, we import the text file into our workspace. To do this we can use open method of python which reads our text file to the workspace.
- Next, we read each word in the input file as a string input. This can be implemented using nested for loop wherein we split each word whenever we encounter a space.
- Next, using regex we remove the punctuations from our input string. This will render token a better syntactic structure and break the sentence bonds.
- Next, we split and store each token into a list using nltk's sentence_tokenize method
- Then finally we remove stop words as we did in previous assignment.

Code here is same as previous one… just that input file is different…

Harshit Mishra (19BCE0799)

a) Input Text File:



Harshit Mishra (19BCE0799)

a) Code Snippets and Output:

In [1]:
```
#Reading input from a text file and saving it as a string
text = ""
with open('test_file2a.txt') as file:
    for line in file:
        for word in line.split():
            text = text + " " + word.lower()
```

In [2]:
```
#Importing libraries
import re
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
```

In [3]:
```
#Removing punctuations from our input file
text = re.sub(r'[^\w\s]', '', text)
text
```

Out[3]:
```
' this is harshit again here i am tokenizing a single line statement'
```

In [4]:
```
#Printing each token
tokens = sent_tokenize(text)
print(tokens)
```

```
[' this is harshit again here i am tokenizing a single line statement']
```

In [5]:
```
#Printing unique tokens
import numpy as np
tokens = np.unique(tokens)
print(tokens)
```

```
[' this is harshit again here i am tokenizing a single line statement']
```

In [6]:
```
#Removing Stopwords
res = []
for x in tokens:
    if x not in set(stopwords.words('english')):
        res.append(x)
```

In [7]:
```
#Printing cleaned tokens in our input text file
print(res)
```

```
[' this is harshit again here i am tokenizing a single line statement']
```

a) Result and output:

1) Input Sentence:

```
This is Harshit again. Here I am tokenizing a single line statement.
```

2) Token without removing stop words:

```
[' this is harshit again here i am tokenizing a single line statement']
```

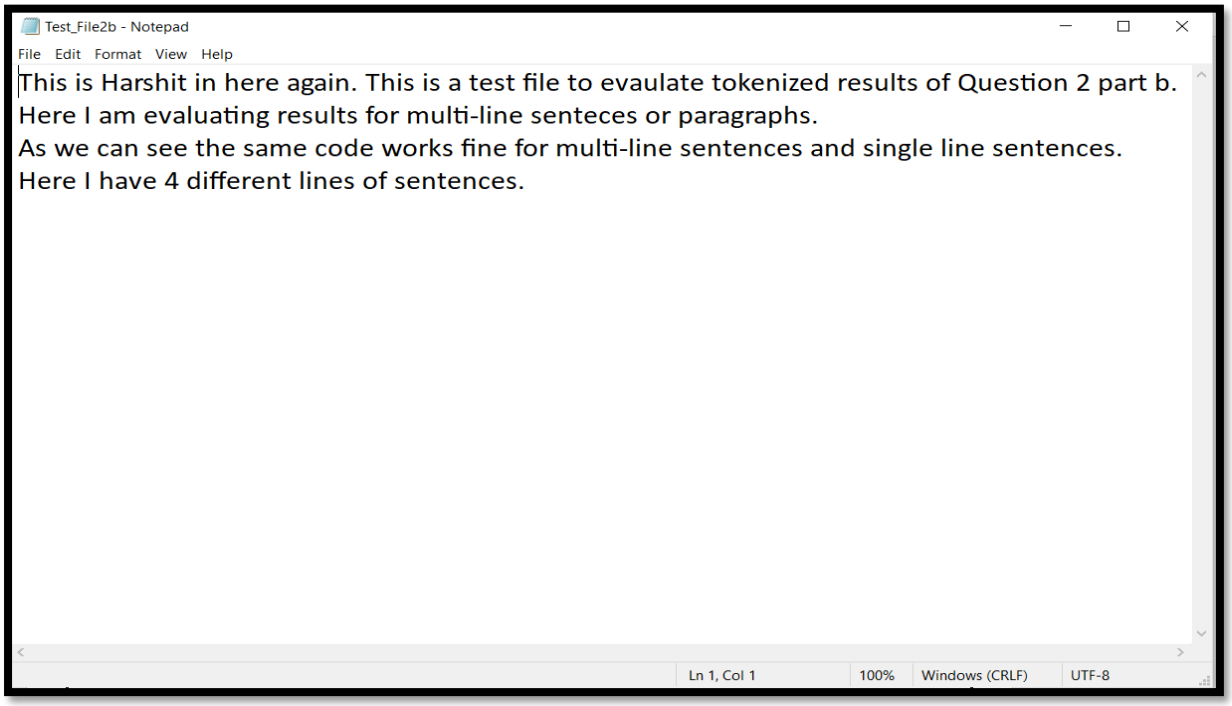
3) Token after removing stop words:

```
[' this is harshit again here i am tokenizing a single line statement']
```

Here we can see that stopwords are not removed. This owes to the fact that sen_tokenize take in complete sentence as a token.

Harshit Mishra (19BCE0799)

b) Input text file:



Harshit Mishra (19BCE0799)

b) Code and Output:

In [1]:
```
#Reading input from a text file and saving it as a string
text = ""
with open('test_file2b.txt') as file:
    for line in file:
        for word in line.split():
            text = text + " " + word.lower()
```

In [2]:
```
#Importing libraries
import re
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
```

In [3]:
```
#Removing punctuations from our input file
text = re.sub(r'[^\w\s]', '', text)
text
```

Out[3]:
```
' this is harshit in here again this is a test file to evaulate tokenized res
ults of question 2 part b here i am evaluating results for multiline senteces
or paragraphs as we can see the same code works fine for multiline sentences
and single line sentences here i have 4 different lines of sentences'
```

In [4]:
```
#Printing each token
tokens = word_tokenize(text)
print(tokens)
```

```
['this', 'is', 'harshit', 'in', 'here', 'again', 'this', 'is', 'a', 'test',
'file', 'to', 'evaulate', 'tokenized', 'results', 'of', 'question', '2', 'par
t', 'b', 'here', 'i', 'am', 'evaluating', 'results', 'for', 'multiline', 'sen
teces', 'or', 'paragraphs', 'as', 'we', 'can', 'see', 'the', 'same', 'code',
'works', 'fine', 'for', 'multiline', 'sentences', 'and', 'single', 'line', 's
entences', 'here', 'i', 'have', '4', 'different', 'lines', 'of', 'sentences']
```

In [5]:
```
#Printing unique tokens
import numpy as np
tokens = np.unique(tokens)
print(tokens)
```

```
['2' '4' 'a' 'again' 'am' 'and' 'as' 'b' 'can' 'code' 'different'
 'evaluating' 'evaulate' 'file' 'fine' 'for' 'harshit' 'have' 'here' 'i'
 'in' 'is' 'line' 'lines' 'multiline' 'of' 'or' 'paragraphs' 'part'
 'question' 'results' 'same' 'see' 'senteces' 'sentences' 'single' 'test'
 'the' 'this' 'to' 'tokenized' 'we' 'works']
```

In [6]:
```
#Removing Stopwords
res = []
for x in tokens:
    if x not in set(stopwords.words('english')):
        res.append(x)
```

In [7]:
```
#Printing cleaned tokens in our input text file
print(res)
```

```
['2', '4', 'b', 'code', 'different', 'evaluating', 'evaulate', 'file', 'fin
e', 'harshit', 'line', 'lines', 'multiline', 'paragraphs', 'part', 'questio
n', 'results', 'see', 'senteces', 'sentences', 'single', 'test', 'tokenized',
'works']
```

Harshit Mishra (19BCE0799)

b) Result and Output:

1) Input Text:

```
This is Harshit in here again. This is a test file to evaulate tokenized results of Question 2 part b.
Here I am evaluating results for multi-line senteces or paragraphs.
As we can see the same code works fine for multi-line sentences and single line sentences.
Here I have 4 different lines of sentences.
```

2) Tokens without removing stop words:

```
['2' '4' 'a' 'again' 'am' 'and' 'as' 'b' 'can' 'code' 'different'
 'evaluating' 'evaulate' 'file' 'fine' 'for' 'harshit' 'have' 'here' 'i'
 'in' 'is' 'line' 'lines' 'multiline' 'of' 'or' 'paragraphs' 'part'
 'question' 'results' 'same' 'see' 'senteces' 'sentences' 'single' 'test'
 'the' 'this' 'to' 'tokenized' 'we' 'works']
```

3) Tokens after removing stop words:

```
['2', '4', 'b', 'code', 'different', 'evaluating', 'evaulate', 'file', 'fin
e', 'harshit', 'line', 'lines', 'multiline', 'paragraphs', 'part', 'questio
n', 'results', 'see', 'senteces', 'sentences', 'single', 'test', 'tokenized',
'works']
```