Name: Krishna Santosh Kabra(27)

Practical 7: Extract Sample document and apply following document preprocessing methods: Tokenization, POS Tagging, stop words removal, Stemming and Lemmatization.

Install or update nltk and BeautifulSoup libraries

```
!pip install nltk -U
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.0)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.2)
```

```
!pip install bs4 -U
```

```
Collecting bs4
  Downloading bs4-0.0.2-py2.py3-none-any.whl (1.2 kB)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from bs4) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->bs4) (2.5)
Installing collected packages: bs4
Successfully installed bs4-0.0.2
```

Download necessary NLTK resources

```
import nltk
```

Download stopwords corpus

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

Download Punkt tokenizer models

```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

Download WordNet corpus

```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

Download averaged perceptron tagger models

```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
True
```

```
import nltk
```

Define a paragraph of text

```
para = 'Rajgad (literal meaning Ruling Fort) is a hill fort situated in Pune district of Maharashtra, India. Formerly known as Murumdev,
```

```
print(para)
```

```
Rajgad (literal meaning Ruling Fort) is a hill fort situated in Pune district of Maharashtra, India. Formerly known as Murumdev, the
```

```
para.split()
```

```
['Rajgad',
 '(literal',
 'meaning',
 'Ruling',
 'Fort)',
 'is',
 'a',
 'hill',
 'fort',
 'situated',
 'in',
 'Pune',
 'district',
 'of',
 'Maharashtra,',
 'India.',
 'Formerly',
 'known',
 'as',
 'Murumdev,',
 'the',
 'fort',
 'was',
 'the',
 'capital',
 'of',
 'the',
 'Maratha',
 'Empire',
 'under',
 'the',
 'rule',
 'of',
 'Chatrapati',
 'Shivaji',
 'maharaj',
 'for',
 'almost',
 '26',
 'years,',
 'after',
 'which',
 'the',
 'capital',
 'was',
 'moved',
 'to',
 'the',
 'Rajgad',
 'Fort.',
 '[1]',
 'Treasures',
 'discovered',
 'from',
 'adjacent',
 'fort',
 'called',
 'Torna',
```

Tokenize the paragraph into sentences

```
from nltk.tokenize import sent_tokenize
```

```
from nltk.tokenize import word_tokenize
```

```
sent = sent_tokenize(para)
```

```
sent[2]
```

```
'[1] Treasures discovered from adjacent fort called Torna were used to completely build and fortify the Rajgad Fort.'
```

Tokenize the paragraph into words

```
words = word_tokenize(para)
```

```
words
```

```
['Rajgad',
 '(',
 'literal',
 'meaning',
 'Ruling',
 'Fort',
 ')',
 'is',
 'a',
 'hill',
 'fort',
 'situated',
 'in',
 'Pune',
 'district',
 'of',
 'Maharashtra',
 ',',
 'India',
 '.',
 'Formerly',
 'known',
 'as',
 'Murumdev',
 ',',
 'the',
 'fort',
 'was',
 'the',
 'capital',
 'of',
 'the',
 'Maratha',
 'Empire',
 'under',
 'the',
 'rule',
 'of',
 'Chatrapati',
 'Shivaji',
 'maharaj',
 'for',
 'almost',
 '26',
 'years',
 ',',
 'after',
 'which',
 'the',
 'capital',
 'was',
 'moved',
 'to',
 'the',
 'Rajgad',
 'Fort',
 '.',
 '[',
```

Load English Stopwords

```
from nltk.corpus import stopwords
```

```
swords = stopwords.words('english')
```

```
swords
```

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
```

```
'he',
'him',
'his',
'himself',
'she',
"she's",
'her',
'hers',
'herself',
'it',
"it's",
'its',
'itself',
'they',
'them',
'their',
'theirs',
'themselves',
'what',
'which',
'who',
'whom',
'this',
'that',
"that'll",
'these',
'those',
'am',
'is',
'are',
'was',
'were',
'be',
'been',
'being',
'have',
'has',
'had',
'having',
'do',
'does'
```

Remove stopwords from the tokenized words

```
x=[word for word in words if word not in swords]
```

```
x
```

```
['Rajgad',
 '(',
 'literal',
 'meaning',
 'Ruling',
 'Fort',
 ')',
 'hill',
 'fort',
 'situated',
 'Pune',
 'district',
 'Maharashtra',
 ',',
 'India',
 '.',
 'Formerly',
 'known',
 'Murumdev',
 ',',
 'fort',
 'capital',
 'Maratha',
 'Empire',
 'rule',
 'Chatrapati',
 'Shivaji',
 'maharaj',
 'almost',
 '26',
 'years',
 ',',
 'capital',
 'moved',
 'Rajgad',
 'Fort',
 '.',
 '[',
 '1',
 ']',
```

```
 'Treasures',
 'discovered',
 'adjacent',
 'fort',
 'called',
 'Torna',
 'used',
 'completely',
 'build',
 'fortify',
 'Rajgad',
 'Fort',
 '.']
```

Apply stemming using PorterStemmer

```
from nltk.stem import PorterStemmer
```

```
ps = PorterStemmer()
```

```
'work'
```

```
'work'
```

```
y = [ps.stem(word) for word in x]
```

```
y
```

```
['rajgad',
 '(',
 'liter',
 'mean',
 'rule',
 'fort',
 ')',
 'hill',
 'fort',
 'situat',
 'pune',
 'district',
 'maharashtra',
 ',',
 'india',
 '.',
 'formerli',
 'known',
 'murumdev',
 ',',
 'fort',
 'capit',
 'maratha',
 'empir',
 'rule',
 'chatrapati',
 'shivaji',
 'maharaj',
 'almost',
 '26',
 'year',
 ',',
 'capit',
 'move',
 'rajgad',
 'fort',
 '.',
 '[',
 '1',
 ']',
 'treasur',
 'discov',
 'adjac',
 'fort',
 'call',
 'torna',
 'use',
 'complet',
 'build',
 'fortifi',
 'rajgad',
 'fort',
 '.']
```

Apply lemmatization using WordNetLemmatizer

```
from nltk.stem import WordNetLemmatizer

wnl = WordNetLemmatizer()

wnl.lemmatize('working', pos ='v')
```

```
'work'
```

```
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
True
```

```
wnl.lemmatize('working',pos ='v')
```

```
'work'
```

```
print(ps.stem('went'))
```

```
went
```

```
print(wnl.lemmatize('went',pos ='v'))
```

```
go
go
```

```
z = [wnl.lemmatize(word,pos='v') for word in x]

z
```

```
['Rajgad',
 '(',
 'literal',
 'mean',
 'Ruling',
 'Fort',
 ')',
 'hill',
 'fort',
 'situate',
 'Pune',
 'district',
 'Maharashtra',
 ',',
 'India',
 '.',
 'Formerly',
 'know',
 'Murumdev',
 ',',
 'fort',
 'capital',
 'Maratha',
 'Empire',
 'rule',
 'Chatrapati',
 'Shivaji',
 'maharaj',
 'almost',
 '26',
 'years',
 ',',
 'capital',
 'move',
 'Rajgad',
 'Fort',
 '.',
 '[',
 '1',
 ']',
 'Treasures',
 'discover',
 'adjacent',
 'fort',
 'call',
 'Torna',
 'use',
 'completely',
 'build',
 'fortify',
```

```
        'Rajgad',
        'Fort',
        '.']
```

Remove punctuation

```
import string
```

```
string.punctuation
```

```
     '!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

```
t = [word for word in words if word not in string.punctuation]
```

```
t
```

```
     ['Rajgad',
      'literal',
      'meaning',
      'Ruling',
      'Fort',
      'is',
      'a',
      'hill',
      'fort',
      'situated',
      'in',
      'Pune',
      'district',
      'of',
      'Maharashtra',
      'India',
      'Formerly',
      'known',
      'as',
      'Murumdev',
      'the',
      'fort',
      'was',
      'the',
      'capital',
      'of',
      'the',
      'Maratha',
      'Empire',
      'under',
      'the',
      'rule',
      'of',
      'Chatrapati',
      'Shivaji',
      'maharaj',
      'for',
      'almost',
      '26',
      'years',
      'after',
      'which',
      'the',
      'capital',
      'was',
      'moved',
      'to',
      'the',
      'Rajgad',
      'Fort',
      '1',
      'Treasures',
      'discovered',
      'from',
      'adjacent',
      'fort',
      'called',
      'Torna',
```

Perform part-of-speech tagging

```
from nltk import pos_tag
```

```
pos_tag(t)
```

```
[('Rajgad', 'NNP'),
 ('literal', 'JJ'),
 ('meaning', 'NN'),
 ('Ruling', 'NNP'),
 ('Fort', 'NNP'),
 ('is', 'VBZ'),
 ('a', 'DT'),
 ('hill', 'NN'),
 ('fort', 'NN'),
 ('situated', 'VBD'),
 ('in', 'IN'),
 ('Pune', 'NNP'),
 ('district', 'NN'),
 ('of', 'IN'),
 ('Maharashtra', 'NNP'),
 ('India', 'NNP'),
 ('Formerly', 'RB'),
 ('known', 'VBN'),
 ('as', 'IN'),
 ('Murumdev', 'NNP'),
 ('the', 'DT'),
 ('fort', 'NN'),
 ('was', 'VBD'),
 ('the', 'DT'),
 ('capital', 'NN'),
 ('of', 'IN'),
 ('the', 'DT'),
 ('Maratha', 'NNP'),
 ('Empire', 'NNP'),
 ('under', 'IN'),
 ('the', 'DT'),
 ('rule', 'NN'),
 ('of', 'IN'),
 ('Chatrapati', 'NNP'),
 ('Shivaji', 'NNP'),
 ('maharaj', 'NN'),
 ('for', 'IN'),
 ('almost', 'RB'),
 ('26', 'CD'),
 ('years', 'NNS'),
 ('after', 'IN'),
 ('which', 'WDT'),
 ('the', 'DT'),
 ('capital', 'NN'),
 ('was', 'VBD'),
 ('moved', 'VBN'),
 ('to', 'TO'),
 ('the', 'DT'),
 ('Rajgad', 'NNP'),
 ('Fort', 'NNP'),
 ('1', 'CD'),
 ('Treasures', 'NNS'),
 ('discovered', 'VBN'),
 ('from', 'IN'),
 ('adjacent', 'JJ'),
 ('fort', 'NN'),
 ('called', 'VBN'),
 ('Torna', 'NNP'),
```

Feature extraction using TF-IDF vectorization

```
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf = TfidfVectorizer()

v = tfidf.fit_transform(t)

v.shape
```

```
(68, 48)
```

Convert the TF-IDF matrix into a DataFrame for visualization

```
import pandas as pd

pd.DataFrame(v)
```

	0
0	(0, 33)\t1.0
1	(0, 24)\t1.0
2	(0, 28)\t1.0
3	(0, 35)\t1.0
4	(0, 16)\t1.0
...	...
63	(0, 4)\t1.0
64	(0, 17)\t1.0
65	(0, 38)\t1.0
66	(0, 33)\t1.0
67	(0, 16)\t1.0

68 rows × 1 columns

Start coding or generate with AI.