dsbdalab 7

March 28, 2024

[1]:
```
#Tokenization
from nltk import word_tokenize, sent_tokenize
sentence1 = "I will walk 500 miles and I would walk 500 more. Just to be the man␣
 ↪who walks a thousand miles to fall down at your door!"
sentence2 = "I played the play playfully as the players were playing in the play␣
 ↪with playfullness"
print('1st Tokenized words:', word_tokenize(sentence1))
print('\n1st Tokenized sentences:', sent_tokenize(sentence1))
print('\n2nd Tokenized words:', word_tokenize(sentence2))
print('\n2nd Tokenized sentences:', sent_tokenize(sentence2))
```

```
1st Tokenized words: ['I', 'will', 'walk', '500', 'miles', 'and', 'I', 'would',
'walk', '500', 'more', '.', 'Just', 'to', 'be', 'the', 'man', 'who', 'walks',
'a', 'thousand', 'miles', 'to', 'fall', 'down', 'at', 'your', 'door', '!']

1st Tokenized sentences: ['I will walk 500 miles and I would walk 500 more.',
'Just to be the man who walks a thousand miles to fall down at your door!']

2nd Tokenized words: ['I', 'played', 'the', 'play', 'playfully', 'as', 'the',
'players', 'were', 'playing', 'in', 'the', 'play', 'with', 'playfullness']

2nd Tokenized sentences: ['I played the play playfully as the players were
playing in the play with playfullness']
```

[2]:
```
#POS Tagging
from nltk import pos_tag
token = word_tokenize(sentence1) + word_tokenize(sentence2)
tagged = pos_tag(token)
print("Tagging Parts of Speech:", tagged)
```

```
Tagging Parts of Speech: [('I', 'PRP'), ('will', 'MD'), ('walk', 'VB'), ('500',
'CD'), ('miles', 'NNS'), ('and', 'CC'), ('I', 'PRP'), ('would', 'MD'), ('walk',
'VB'), ('500', 'CD'), ('more', 'JJR'), ('.', '.'), ('Just', 'NNP'), ('to',
'TO'), ('be', 'VB'), ('the', 'DT'), ('man', 'NN'), ('who', 'WP'), ('walks',
'VBZ'), ('a', 'DT'), ('thousand', 'NN'), ('miles', 'NNS'), ('to', 'TO'),
('fall', 'VB'), ('down', 'RP'), ('at', 'IN'), ('your', 'PRP$'), ('door', 'NN'),
('!', '.'), ('I', 'PRP'), ('played', 'VBD'), ('the', 'DT'), ('play', 'NN'),
('playfully', 'RB'), ('as', 'IN'), ('the', 'DT'), ('players', 'NNS'), ('were',
```

```
'VBD'), ('playing', 'VBG'), ('in', 'IN'), ('the', 'DT'), ('play', 'NN'),
('with', 'IN'), ('playfullness', 'NN')]
```

[3]:
```
#Stop-Words Removal 1
from nltk.corpus import stopwords
stop_words=set(stopwords.words("english"))
print(stop_words)
token = word_tokenize(sentence1)
cleaned_token = []
for word in token:
    if word not in stop_words:
        cleaned_token.append(word)
print('Token Sentence:', token)
print('\nCleaned version:', cleaned_token)
```

```
{'did', 'they', 'an', "wasn't", 'over', 'now', 'out', 'themselves', 'each',
'he', 'this', 'why', 'other', 'such', 'below', "hadn't", 'she', 'am', 'most',
"don't", 'had', 'haven', "you'd", 'wasn', 'should', 'yourselves', 'if', 'has',
'can', 'from', 'myself', 'been', 'the', 'while', 'because', 'there', 'have',
'doing', 'which', 'all', 'above', 'its', 'ourselves', 's', "needn't", 've',
'between', 'where', 'that', "that'll", 'to', 'whom', 'against', 'after', 'll',
'how', 'no', 'any', "won't", "weren't", "mightn't", "couldn't", 'couldn', 'm',
'i', 'd', "aren't", 'mustn', 'in', 'you', 'them', 'of', "should've", 'shan',
'ain', 'shouldn', 'only', 'having', 'same', 'our', 'herself', 'here', 'and',
'won', 'do', "you'll", 'so', 'very', 'ours', 'when', 'mightn', 'his', 'own',
'under', 'himself', 'as', 'during', "you've", 't', "wouldn't", 'these', 'with',
'will', "doesn't", 'her', 'isn', "it's", 'don', 'were', 'doesn', "you're",
"haven't", 'yourself', 'aren', 'me', 'wouldn', 'few', 'a', 'nor', 're', 'yours',
'does', 'through', 'being', 'or', "didn't", 'into', 'both', "hasn't", 'my',
'again', 'their', 'too', 'y', 'was', 'ma', 'at', 'further', 'didn', 'until',
'hadn', 'hers', 'what', 'who', 'your', 'those', "she's", 'before', 'be', 'it',
'itself', 'once', "mustn't", 'by', 'on', 'hasn', "isn't", 'than', 'up', 'are',
'o', 'more', 'needn', 'down', 'just', 'him', 'but', 'theirs', "shouldn't",
'not', 'then', 'some', 'off', 'we', 'for', 'is', 'weren', 'about', "shan't"}
Token Sentence: ['I', 'will', 'walk', '500', 'miles', 'and', 'I', 'would',
'walk', '500', 'more', '.', 'Just', 'to', 'be', 'the', 'man', 'who', 'walks',
'a', 'thousand', 'miles', 'to', 'fall', 'down', 'at', 'your', 'door', '!']

Cleaned version: ['I', 'walk', '500', 'miles', 'I', 'would', 'walk', '500', '.',
'Just', 'man', 'walks', 'thousand', 'miles', 'fall', 'door', '!']
```

[4]:
```
#Stop-Words Removal 2
from nltk.corpus import stopwords
stop_words=set(stopwords.words("english"))
print(stop_words)
token = word_tokenize(sentence2)
cleaned_token = []
for word in token:
```

```
    if word not in stop_words:
        cleaned_token.append(word)
print('Token Sentence:', token)
print('\nCleaned version:', cleaned_token)
```

```
{'did', 'they', 'an', "wasn't", 'over', 'now', 'out', 'themselves', 'each',
'he', 'this', 'why', 'other', 'such', 'below', "hadn't", 'she', 'am', 'most',
"don't", 'had', 'haven', "you'd", 'wasn', 'should', 'yourselves', 'if', 'has',
'can', 'from', 'myself', 'been', 'the', 'while', 'because', 'there', 'have',
'doing', 'which', 'all', 'above', 'its', 'ourselves', 's', "needn't", 've',
'between', 'where', 'that', "that'll", 'to', 'whom', 'against', 'after', 'll',
'how', 'no', 'any', "won't", "weren't", "mightn't", "couldn't", 'couldn', 'm',
'i', 'd', "aren't", 'mustn', 'in', 'you', 'them', 'of', "should've", 'shan',
'ain', 'shouldn', 'only', 'having', 'same', 'our', 'herself', 'here', 'and',
'won', 'do', "you'll", 'so', 'very', 'ours', 'when', 'mightn', 'his', 'own',
'under', 'himself', 'as', 'during', "you've", 't', "wouldn't", 'these', 'with',
'will', "doesn't", 'her', 'isn', "it's", 'don', 'were', 'doesn', "you're",
"haven't", 'yourself', 'aren', 'me', 'wouldn', 'few', 'a', 'nor', 're', 'yours',
'does', 'through', 'being', 'or', "didn't", 'into', 'both', "hasn't", 'my',
'again', 'their', 'too', 'y', 'was', 'ma', 'at', 'further', 'didn', 'until',
'hadn', 'hers', 'what', 'who', 'your', 'those', "she's", 'before', 'be', 'it',
'itself', 'once', "mustn't", 'by', 'on', 'hasn', "isn't", 'than', 'up', 'are',
'o', 'more', 'needn', 'down', 'just', 'him', 'but', 'theirs', "shouldn't",
'not', 'then', 'some', 'off', 'we', 'for', 'is', 'weren', 'about', "shan't"}
Token Sentence: ['I', 'played', 'the', 'play', 'playfully', 'as', 'the',
'players', 'were', 'playing', 'in', 'the', 'play', 'with', 'playfullness']

Cleaned version: ['I', 'played', 'play', 'playfully', 'players', 'playing',
'play', 'playfullness']
```

[5]:
```
#Stemming 1
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
token = word_tokenize(sentence1)
stemmed = [stemmer.stem(word) for word in token]
print(" ".join(stemmed))
```

```
i will walk 500 mile and i would walk 500 more . just to be the man who walk a
thousand mile to fall down at your door !
```

[6]:
```
#Lemmatization 1
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
token = word_tokenize(sentence1)
lemmatized_output = [lemmatizer.lemmatize(word) for word in token]
print(" ".join(lemmatized_output))
```

```
I will walk 500 mile and I would walk 500 more . Just to be the man who walk a
thousand mile to fall down at your door !
```

[7]:
```
#Stemming 2
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
token = word_tokenize(sentence2)
stemmed = [stemmer.stem(word) for word in token]
print(" ".join(stemmed))
```

```
i play the play play as the player were play in the play with playful
```

[8]:
```
#Lemmatization 2
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
token = word_tokenize(sentence2)
lemmatized_output = [lemmatizer.lemmatize(word) for word in token]
print(" ".join(lemmatized_output))
```

```
I played the play playfully a the player were playing in the play with
playfullness
```

[9]:
```
# Instantiate the TfidfVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
data = {'text': ['I will walk 500 miles and I would walk 500 more. Just to be␣
 ↪the man who walks a thousand miles to fall down at your door!']}
tf = TfidfVectorizer()
text_tf = tf.fit_transform(data['text'])
print(text_tf)
```

```
  (0, 4)        0.17677669529663687
  (0, 19)       0.17677669529663687
  (0, 2)        0.17677669529663687
  (0, 5)        0.17677669529663687
  (0, 6)        0.17677669529663687
  (0, 12)       0.17677669529663687
  (0, 15)       0.17677669529663687
  (0, 16)       0.17677669529663687
  (0, 8)        0.17677669529663687
  (0, 11)       0.17677669529663687
  (0, 3)        0.17677669529663687
  (0, 13)       0.35355339059327373
  (0, 7)        0.17677669529663687
  (0, 10)       0.17677669529663687
  (0, 18)       0.17677669529663687
  (0, 1)        0.17677669529663687
  (0, 9)        0.35355339059327373
  (0, 0)        0.35355339059327373
  (0, 14)       0.35355339059327373
  (0, 17)       0.17677669529663687
```

[ ]: