practical-07

April 26, 2024

[1]:
```
import nltk
from nltk.tokenize import word_tokenize,sent_tokenize
from nltk.corpus import stopwords
```

[2]:
```
text = "Python is a high-level, general-purpose programming language. Its␣
 ↪design philosophy emphasizes code readability with the use of significant␣
 ↪indentation. Python is dynamically typed and garbage-collected. It supports␣
 ↪multiple programming paradigms."
```

[3]:
```
# Tokenization
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\gugal\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[3]:
```
True
```

[4]:
```
sent_tokenize(text)
```

[4]:
```
['Python is a high-level, general-purpose programming language.',
 'Its design philosophy emphasizes code readability with the use of significant
indentation.',
 'Python is dynamically typed and garbage-collected.',
 'It supports multiple programming paradigms.']
```

[5]:
```
tokens = word_tokenize(text)
tokens
```

[5]:
```
['Python',
 'is',
 'a',
 'high-level',
 ',',
 'general-purpose',
 'programming',
 'language',
 '.',
```

```
 'Its',
 'design',
 'philosophy',
 'emphasizes',
 'code',
 'readability',
 'with',
 'the',
 'use',
 'of',
 'significant',
 'indentation',
 '.',
 'Python',
 'is',
 'dynamically',
 'typed',
 'and',
 'garbage-collected',
 '.',
 'It',
 'supports',
 'multiple',
 'programming',
 'paradigms',
 '.']
```

[6]:
```
nltk.download("stopwords")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\gugal\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[6]: True

[7]:
```
stopwords = set(stopwords.words("english"))
print(stopwords)
```

```
{'mightn', 'being', 'ours', 'didn', 'during', "needn't", 'our', 'these',
"wasn't", 'so', 'now', "hadn't", 'how', 'itself', 'do', 'who', 'ma', 'theirs',
'just', 'with', 'whom', 're', 'but', 'nor', 'own', 'me', 'aren', "hasn't",
"you're", 'had', 'as', 'couldn', "didn't", 'your', "shouldn't", 'until', 'up',
'what', 'very', 'at', 'where', 'hers', 'same', 'be', 'should', 'those', 'am',
'both', "you've", 'which', 'under', 'he', 'against', 'than', 'hadn', 've',
'yourselves', 'of', 'its', 'his', "it's", 'are', 'only', 'yourself', 'shan',
'down', 'over', 'having', 'when', 'such', 'myself', 'by', 'below', 'o', 'most',
"couldn't", 'to', 'doesn', "isn't", 'and', 'has', 'doing', 'out', "mightn't",
"weren't", 'were', 'we', 'a', "she's", "wouldn't", 'you', 'there', 'can',
```

```
'them', 'in', 'on', 'him', 'i', 'each', 'ourselves', 'through', 'needn', 'for',
'will', 'm', 'from', 'was', 'don', 'hasn', "aren't", 'have', "don't", 'they',
"you'd", 'been', 'further', 'any', 'themselves', "mustn't", 't', 'the', 'this',
'while', 'because', 'does', 'that', 'before', 'once', 'haven', 'then', 'too',
'again', 's', 'or', 'if', "shan't", 'ain', 'yours', 'no', 'above', 'her', 'few',
"should've", 'wouldn', 'shouldn', 'more', "haven't", 'after', 'she', 'll',
'into', 'off', 'is', "you'll", 'mustn', "won't", 'won', 'weren', 'other',
"that'll", 'did', 'd', "doesn't", 'it', 'why', 'wasn', 'between', 'y', 'about',
'my', 'herself', 'their', 'some', 'not', 'isn', 'here', 'himself', 'an', 'all'}
```

[8]:
```
filtered_list = []
for i in tokens:
    if i.lower() not in stopwords:
        filtered_list.append(i)
filtered_list
```

[8]:
```
['Python',
 'high-level',
 ',',
 'general-purpose',
 'programming',
 'language',
 '.',
 'design',
 'philosophy',
 'emphasizes',
 'code',
 'readability',
 'use',
 'significant',
 'indentation',
 '.',
 'Python',
 'dynamically',
 'typed',
 'garbage-collected',
 '.',
 'supports',
 'multiple',
 'programming',
 'paradigms',
 '.']
```

[9]:
```
from nltk.stem import PorterStemmer
stemmer = PorterStemmer()
```

[10]:
```
Stemming = ["running", "jumps", "happily", "discovery",␣
 ↪"discoveries","good","corpora"]
After_stemming = [stemmer.stem(word) for word in Stemming]
After_stemming
```

[10]:
```
['run', 'jump', 'happili', 'discoveri', 'discoveri', 'good', 'corpora']
```

[11]:
```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\gugal\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[11]:
```
True
```

[12]:
```
from nltk.stem import WordNetLemmatizer
lemmed = [WordNetLemmatizer().lemmatize(i) for i in Stemming]
lemmed
```

[12]:
```
['running', 'jump', 'happily', 'discovery', 'discovery', 'good', 'corpus']
```

[13]:
```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\gugal\AppData\Roaming\nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

[13]:
```
True
```

[14]:
```
nltk.pos_tag(tokens)
```

[14]:
```
[('Python', 'NNP'),
 ('is', 'VBZ'),
 ('a', 'DT'),
 ('high-level', 'JJ'),
 (',', ','),
 ('general-purpose', 'JJ'),
 ('programming', 'NN'),
 ('language', 'NN'),
 ('.', '.'),
 ('Its', 'PRP$'),
 ('design', 'NN'),
 ('philosophy', 'NN'),
 ('emphasizes', 'VBZ'),
 ('code', 'JJ'),
 ('readability', 'NN'),
```

```
 ('with', 'IN'),
 ('the', 'DT'),
 ('use', 'NN'),
 ('of', 'IN'),
 ('significant', 'JJ'),
 ('indentation', 'NN'),
 ('.', '.'),
 ('Python', 'NNP'),
 ('is', 'VBZ'),
 ('dynamically', 'RB'),
 ('typed', 'JJ'),
 ('and', 'CC'),
 ('garbage-collected', 'JJ'),
 ('.', '.'),
 ('It', 'PRP'),
 ('supports', 'VBZ'),
 ('multiple', 'JJ'),
 ('programming', 'VBG'),
 ('paradigms', 'NN'),
 ('.', '.')]
```

[15]:
```
nltk.download('tagsets')
```

```
[nltk_data] Downloading package tagsets to
[nltk_data]     C:\Users\gugal\AppData\Roaming\nltk_data…
[nltk_data]   Package tagsets is already up-to-date!
```

[15]: True

[16]:
```
nltk.help.upenn_tagset()
```

```
$: dollar
    $ -$ --$ A$ C$ HK$ M$ NZ$ S$ U.S.$ US$
'': closing quotation mark
    ' ''
(: opening parenthesis
    ( [ {
): closing parenthesis
    ) ] }
,: comma
    ,
--: dash
    --
.: sentence terminator
    . ! ?
:: colon or ellipsis
    : ; …
CC: conjunction, coordinating
```

& 'n and both but either et for less minus neither nor or plus so therefore times v. versus vs. whether yet

CD: numeral, cardinal
mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one forty-seven 1987 twenty '79 zero two 78-degrees eighty-four IX '60s .025 fifteen 271,124 dozen quintillion DM2,000 …

DT: determiner
all an another any both del each either every half la many much nary neither no some such that the them these this those

EX: existential there
there

FW: foreign word
gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous lutihaw alai je jour objets salutaris fille quibusdam pas trop Monte terram fiche oui corporis …

IN: preposition or conjunction, subordinating
astride among uppon whether out inside pro despite on by throughout below within for towards near behind atop around if like until below next into if beside …

JJ: adjective or numeral, ordinal
third ill-mannered pre-war regrettable oiled calamitous first separable ectoplasmic battery-powered participatory fourth still-to-be-named multilingual multi-disciplinary …

JJR: adjective, comparative
bleaker braver breezier briefer brighter brisker broader bumper busier calmer cheaper choosier cleaner clearer closer colder commoner costlier cozier creamier crunchier cuter …

JJS: adjective, superlative
calmest cheapest choicest classiest cleanest clearest closest commonest corniest costliest crassest creepiest crudest cutest darkest deadliest dearest deepest densest dinkiest …

LS: list item marker
A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002 SP-44005 SP-44007 Second Third Three Two * a b c d first five four one six three two

MD: modal auxiliary
can cannot could couldn't dare may might must need ought shall should shouldn't will would

NN: noun, common, singular or mass
common-carrier cabbage knuckle-duster Casino afghan shed thermostat investment slide humour falloff slick wind hyena override subhumanity machinist …

NNP: noun, proper, singular
Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA Shannon A.K.C. Meltex Liverpool …

NNPS: noun, proper, plural
Americans Americas Amharas Amityvilles Amusements Anarcho-Syndicalists

Andalusians Andes Andruses Angels Animals Anthony Antilles Antiques Apache Apaches Apocrypha …

NNS: noun, common, plural

undergraduates scotches bric-a-brac products bodyguards facets coasts divestitures storehouses designs clubs fragrances averages subjectivists apprehensions muses factory-jobs …

PDT: pre-determiner

all both half many quite such sure this

POS: genitive marker

' 's

PRP: pronoun, personal

hers herself him himself hisself it itself me myself one oneself ours ourselves ownself self she thee theirs them themselves they thou thy us

PRP$: pronoun, possessive

her his mine my our ours their thy your

RB: adverb

occasionally unabatingly maddeningly adventurously professedly stirringly prominently technologically magisterially predominately swiftly fiscally pitilessly …

RBR: adverb, comparative

further gloomier grander graver greater grimmer harder harsher healthier heavier higher however larger later leaner lengthier less-perfectly lesser lonelier longer louder lower more …

RBS: adverb, superlative

best biggest bluntest earliest farthest first furthest hardest heartiest highest largest least less most nearest second tightest worst

RP: particle

aboard about across along apart around aside at away back before behind by crop down ever fast for forth from go high i.e. in into just later low more off on open out over per pie raising start teeth that through under unto up up-pp upon whole with you

SYM: symbol

% & ' '' ''.)). * + ,. < = > @ A[fj] U.S U.S.S.R * ** ***

TO: "to" as preposition or infinitive marker

to

UH: interjection

Goodbye Goody Gosh Wow Jeepers Jee-sus Hubba Hey Kee-reist Oops amen huh howdy uh dammit whammo shucks heck anyways whodunnit honey golly man baby diddle hush sonuvabitch …

VB: verb, base form

ask assemble assess assign assume atone attention avoid bake balkanize bank begin behold believe bend benefit bevel beware bless boil bomb boost brace break bring broil brush build …

VBD: verb, past tense

dipped pleaded swiped regummed soaked tidied convened halted registered cushioned exacted snubbed strode aimed adopted belied figgered speculated wore appreciated contemplated …

VBG: verb, present participle or gerund

```
    telegraphing stirring focusing angering judging stalling lactating
    hankerin' alleging veering capping approaching traveling besieging
    encrypting interrupting erasing wincing …
VBN: verb, past participle
    multihulled dilapidated aerosolized chaired languished panelized used
    experimented flourished imitated reunifed factored condensed sheared
    unsettled primed dubbed desired …
VBP: verb, present tense, not 3rd person singular
    predominate wrap resort sue twist spill cure lengthen brush terminate
    appear tend stray glisten obtain comprise detest tease attract
    emphasize mold postpone sever return wag …
VBZ: verb, present tense, 3rd person singular
    bases reconstructs marks mixes displeases seals carps weaves snatches
    slumps stretches authorizes smolders pictures emerges stockpiles
    seduces fizzes uses bolsters slaps speaks pleads …
WDT: WH-determiner
    that what whatever which whichever
WP: WH-pronoun
    that what whatever whatsoever which who whom whosoever
WP$: WH-pronoun, possessive
    whose
WRB: Wh-adverb
    how however whence whenever where whereby whereever wherein whereof why
``: opening quotation mark
    ` ``
```

[17]:
```
# Create representation of documents by calculating Term Frequency and Inverse␣
↪DocumentFrequency.
```

[22]:
```
Document_A = "Jupiter is the largest planet."
Document_B = "Mars is the fourth planet from the sun."

tokens_A = word_tokenize(Document_A.lower())
tokens_B = word_tokenize(Document_B.lower())
count = 0
occur_word_A = count(tokens_A)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[22], line 7
      5 tokens_B = word_tokenize(Document_B.lower())
      6 count = 0
----> 7 occur_word_A = count(tokens_A)

TypeError: 'int' object is not callable
```

[23]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

# Sample documents
document_A = "Jupiter is the largest planet."
document_B = "Mars is the fourth planet from the sun."

# Create a list of documents
documents = [document_A, document_B]

# Calculate Term Frequency (TF)
tf_vectorizer = TfidfVectorizer(use_idf=False)  # Use_idf=False for TF␣
 ↪calculation
tf_matrix = tf_vectorizer.fit_transform(documents)

# Display Term Frequency (TF) representation
print("Term Frequency (TF) Matrix:")
print(tf_matrix.toarray())

# Calculate Inverse Document Frequency (IDF)
idf_vectorizer = TfidfVectorizer(use_idf=True)  # Use_idf=True for IDF␣
 ↪calculation
idf_matrix = idf_vectorizer.fit_transform(documents)

# Display Inverse Document Frequency (IDF) representation
print("\nInverse Document Frequency (IDF) Matrix:")
print(idf_matrix.toarray())

# Calculate TF-IDF
tfidf_vectorizer = TfidfVectorizer(use_idf=True)
tfidf_matrix = tfidf_vectorizer.fit_transform(documents)

# Display TF-IDF representation
print("\nTF-IDF Matrix:")
print(tfidf_matrix.toarray())
```

```
Term Frequency (TF) Matrix:
[[0.         0.         0.4472136  0.4472136  0.4472136  0.
  0.4472136  0.         0.4472136 ]
 [0.31622777 0.31622777 0.31622777 0.         0.         0.31622777
  0.31622777 0.31622777 0.63245553]]

Inverse Document Frequency (IDF) Matrix:
[[0.         0.         0.37930349 0.53309782 0.53309782 0.
  0.37930349 0.         0.37930349]
 [0.37695709 0.37695709 0.26820807 0.         0.         0.37695709
  0.26820807 0.37695709 0.53641614]]
```

```
TF-IDF Matrix:
[[0.         0.         0.37930349 0.53309782 0.53309782 0.
  0.37930349 0.         0.37930349]
 [0.37695709 0.37695709 0.26820807 0.         0.         0.37695709
  0.26820807 0.37695709 0.53641614]]
```

[24]:
```python
from sklearn.feature_extraction.text import TfidfVectorizer

# Sample documents
document_A = "Jupiter is the largest planet."
document_B = "Mars is the fourth planet from the sun."

# Create a list of documents
documents = [document_A, document_B]

# Calculate Term Frequency (TF)
tf_vectorizer = TfidfVectorizer(use_idf=False)  # Use_idf=False for TF␣
 ↪calculation
tf_matrix = tf_vectorizer.fit_transform(documents)

# Get feature names (words)
feature_names = tf_vectorizer.get_feature_names_out()

# Display Term Frequency (TF) representation with words
print("Term Frequency (TF) Matrix:")
for i, doc in enumerate(documents):
    print("Document {}: ".format(i+1))
    for word, tf_value in zip(feature_names, tf_matrix.toarray()[i]):
        print("{}: {:.4f}".format(word, tf_value))
    print()

# Calculate Inverse Document Frequency (IDF)
idf_vectorizer = TfidfVectorizer(use_idf=True)  # Use_idf=True for IDF␣
 ↪calculation
idf_matrix = idf_vectorizer.fit_transform(documents)

# Display Inverse Document Frequency (IDF) representation with words
print("\nInverse Document Frequency (IDF) Matrix:")
for word, idf_value in zip(feature_names, idf_matrix.toarray()[0]):
    print("{}: {:.4f}".format(word, idf_value))
print()

# Calculate TF-IDF
tfidf_vectorizer = TfidfVectorizer(use_idf=True)
tfidf_matrix = tfidf_vectorizer.fit_transform(documents)

# Display TF-IDF representation with words
```

```
print("\nTF-IDF Matrix:")
for i, doc in enumerate(documents):
    print("Document {}: ".format(i+1))
    for word, tfidf_value in zip(feature_names, tfidf_matrix.toarray()[i]):
        print("{}: {:.4f}".format(word, tfidf_value))
    print()
```

```
Term Frequency (TF) Matrix:
Document 1:
fourth: 0.0000
from: 0.0000
is: 0.4472
jupiter: 0.4472
largest: 0.4472
mars: 0.0000
planet: 0.4472
sun: 0.0000
the: 0.4472

Document 2:
fourth: 0.3162
from: 0.3162
is: 0.3162
jupiter: 0.0000
largest: 0.0000
mars: 0.3162
planet: 0.3162
sun: 0.3162
the: 0.6325


Inverse Document Frequency (IDF) Matrix:
fourth: 0.0000
from: 0.0000
is: 0.3793
jupiter: 0.5331
largest: 0.5331
mars: 0.0000
planet: 0.3793
sun: 0.0000
the: 0.3793


TF-IDF Matrix:
Document 1:
fourth: 0.0000
from: 0.0000
```

```
is: 0.3793
jupiter: 0.5331
largest: 0.5331
mars: 0.0000
planet: 0.3793
sun: 0.0000
the: 0.3793

Document 2:
fourth: 0.3770
from: 0.3770
is: 0.2682
jupiter: 0.0000
largest: 0.0000
mars: 0.3770
planet: 0.2682
sun: 0.3770
the: 0.5364
```