Page No.
Date

Name:- Rohini Janardan Devkar
PRN no:- 72030818G
Roll no:- 23272
Class :- TE2(comp)
DSBDA Lab

Practical No. 7
Text Analytics

* Aim:- 1) Extract sample document and apply following document preprocessing methods:- Tokenization, POS Tagging, stop words removal, Stemming and lemmitization.
2) Create representation of document by calculating Term Frequency and Inverse Document Frequency.

Theory:-

Text analysis (TA) is a machine learning technique used to automatically extract valuable insights from unstructured text data. Companies use text analysis tools to quickly digest online data and documents, and transform them into actionable insights. Natural language processing is one of the components of text analysis. NLP helps identified sentiment, finding entities in the sentence, and category of blog/article.

Text analysis operations using NLTK:-

NLTK is a powerful python package that provide a set of diverse natural language algorithms. It is free, open source, easy to use, large community and well documented. NLTK consist of the most common algorithms seen as tokenizing, part of speech tagging, stemming, sentiment analysis topic segmentation and named entry recognition.

Page No.
Date

Tokenization :-

Tokenization is the first step analysis. The process of breaking down a text paragraph into smaller chunks as words or sentence is called tokenization.

POS tagging :-

The primary target of part-of-speech (POS) tagging is to identify the grammatical group of a given word whether it is a noun, pronoun, adjective, verb, adverb, etc. PoS tagging looks for relationship within the sentence and assigns a corresponding tag to the world.

Stop words removal :-

Stop words considered as noise in the text. Test may contain stop words such as is, am, are, this, a, an, the, etc. In NLTk, for removing stopwords you need to create a list of stopwords and filter our list of tokens from these words.

Stemming :-

Stemming is a process of linguistic normalization, which reduce words to their word root word or chops off the derivations affixes.

Lemmatization :-

Lemmatization reduces words to their base word, which is linguistically correct lemmas. It transforms root word with the use of vocabulary and morphological analysis. Lemmatization is usually more sophisticated than stemming.

Page No.
Date

Term Frequency :-

Term Frequency means how often a term occurs in a document. In the context of natural language term, correspond to words or phrase.

$$TF(t,d) = \frac{\text{count of } t \text{ in } d}{\text{no of words in } d}$$

Inverse Term Frequency :-

The inverse document frequency is a measures of whatever of term is common or rare in given document corpus. It is obtained by dividing the total no. of documents by the number of documents containing the term in the corpus.

$$idf(t) = \frac{N}{df}$$

Data Science And Big Data Analytics Practical 7

==========

Name:- Rohini Janardan Devkar

Roll no:- 23272

PRN no:- 72030818G

Class :- TE2(COMP)

==========

Text Analytics

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

Create representation of document by calculating Term Frequency and Inverse Document Frequency.

==========

In [2]:
```
import nltk
from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('omw-1.4')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Lenovo\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[2]:
```
True
```

In [3]:
```
text="""Hello Mr. Smith, how are you doing today? The weather is great, and city
The sky is pinkish-blue. You shouldn't eat cardboard"""
```

In [4]:
```
#Sentence Tokenization
tokenized_text=sent_tokenize(text)
print("\n#Sentence Tokenization")
print(tokenized_text)
```

```
#Sentence Tokenization
['Hello Mr. Smith, how are you doing today?', 'The weather is great, and city i
s awesome.', 'The sky is pinkish-blue.', "You shouldn't eat cardboard"]
```

In [5]:
```
#Word Tokenization
tokenized_word=word_tokenize(text)
print("\n#Word Tokenization")
print(tokenized_word)
```

```
#Word Tokenization
['Hello', 'Mr.', 'Smith', ',', 'how', 'are', 'you', 'doing', 'today', '?', 'Th
e', 'weather', 'is', 'great', ',', 'and', 'city', 'is', 'awesome', '.', 'The',
'sky', 'is', 'pinkish-blue', '.', 'You', 'should', "n't", 'eat', 'cardboard']
```

In [6]:
```
#Removing Stopwords
print("\n#Removing Stopwords")
stop_words=set(stopwords.words("english"))
filtered_sent=[]
for w in tokenized_word:
    if w not in stop_words:
        filtered_sent.append(w)
print("Tokenized Sentence:",tokenized_word)
print("\nFilterd Sentence:",filtered_sent)
```

```
#Removing Stopwords
Tokenized Sentence: ['Hello', 'Mr.', 'Smith', ',', 'how', 'are', 'you', 'doin
g', 'today', '?', 'The', 'weather', 'is', 'great', ',', 'and', 'city', 'is', 'a
wesome', '.', 'The', 'sky', 'is', 'pinkish-blue', '.', 'You', 'should', "n't",
'eat', 'cardboard']

Filterd Sentence: ['Hello', 'Mr.', 'Smith', ',', 'today', '?', 'The', 'weathe
r', 'great', ',', 'city', 'awesome', '.', 'The', 'sky', 'pinkish-blue', '.', 'Y
ou', "n't", 'eat', 'cardboard']
```

In [7]:
```
#Stemming
ps =PorterStemmer()
stemmed_words=[]
for w in tokenized_word:
    stemmed_words.append(ps.stem(w))
print("\nStemmed Sentence:",stemmed_words)
```

```
Stemmed Sentence: ['hello', 'mr.', 'smith', ',', 'how', 'are', 'you', 'do', 'to
day', '?', 'the', 'weather', 'is', 'great', ',', 'and', 'citi', 'is', 'awesom',
'.', 'the', 'sky', 'is', 'pinkish-blu', '.', 'you', 'should', "n't", 'eat', 'ca
rdboard']
```

In [8]:
```
#Lemmatization
lemmed_words = []
lem = WordNetLemmatizer()
for w in tokenized_word:
    lemmed_words.append(lem.lemmatize(w))
print("\nLemmatized Sentence:",lemmed_words)
```

```
Lemmatized Sentence: ['Hello', 'Mr.', 'Smith', ',', 'how', 'are', 'you', 'doin
g', 'today', '?', 'The', 'weather', 'is', 'great', ',', 'and', 'city', 'is', 'a
wesome', '.', 'The', 'sky', 'is', 'pinkish-blue', '.', 'You', 'should', "n't",
'eat', 'cardboard']
```

In [9]:
```
#POS Tagging
sent = "Albert Einstein was born in Ulm, Germany in 1879."
tokens=nltk.word_tokenize(sent)
pos_tagging = nltk.pos_tag(tokens)
print("\nPOS Tagging:",pos_tagging)
```

```
POS Tagging: [('Albert', 'NNP'), ('Einstein', 'NNP'), ('was', 'VBD'), ('born',
'VBN'), ('in', 'IN'), ('Ulm', 'NNP'), (',', ','), ('Germany', 'NNP'), ('in', 'I
N'), ('1879', 'CD'), ('.', '.')]
```