Piyusha · Supe.

Experiment - 07. (Group A)

Title of the Assignment : Text analytics.

Problem Statement :

1. Extract sample document and apply following document preprocessing methods : Tokenization, Pos Tagging , stop words removal , Stemming and lemmatization.
2. Create representation of document by calculating Term frequency and Inverse document frequency.

Objective of the assignment - Students should be able to perform Text Analysis using TF , IDF algorithm.

Pre - requisite : 1. Basic of python.
2. Basic of english language.

THEORY :

1] Basic concepts of text analytics :-

- Text mining is also referred to as text analytics. Text mining is a process of exploring sizable textual data and finding patterns.
- Text mining processes the text itself while NLP processes with underlying meta data.
- Finding frequency, count of words , length of sentence presence / absence of words is called text mining.

2] • Text Analysis operations using natural language toolkit.

- NLTK (natural language toolkit) is a leading platform for building python programs to work with human language data.
- It provides easy to use interfaces and lexical resources such as wordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning.
- Analysing movie reviews is one of the classic examples to demonstrate a simple NLP Bag of words model, on movie reviews.

2.1] Tokenization:

- Tokenization is the first step in text analytics The process of breaking down a text paragraph into smaller chunks such as words or sentences is called tokenization.
- Sentence tokenization - split a paragraph into list of sentences using sent-tokenize() method.
- Word tokenization - Split a sentence into list of words using word tokenize method.

2.2] Stop words removal.

Stop words considered as noise in the text. Text 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 out on your list of tokens from these words.

2.3] Stemming and Lemmatization -

- Stemming - is a normalization technique where lists of tokenized words are converted into shortened root words to remove redundancy. Stemming is a process of reducing inflected (or sometimes derived) words to their word stem base or root form.

- Lemmatization - in NLTK is algorithmic process of finding lemma of a word depending on its meaning and context. It usually refers to the morphological analysis of words, which aims to remove inflectional endings. It helps in returning the base or dictionary form of a word known as the lemma.

  tg: lemma for studies is study.

2.4] POS Tagging.

- POS (Parts of speech) tagging tell us about grammatical information of words of the sentence by assigning specific token (Determiner, noun, adjective, adverb, verb, etc). As tag to each words.

- Words can have one or more pos depending upon context where it is used.

3) Text Analysis using TF - IDF -

Term frequency inverse document frequency is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.

- **Term frequency (TF) -**

It is a measure of the frequency of a word in a document (d). TF is defined as the ratio of a words occurence in a document to the total number of words in a document.

$$TF = (w,d) = \frac{\text{occurrences of } w \text{ in a document } d}{\text{total number of words in doc } d.}$$

Example -

Document	Text.	Total no. of words.
A	Jupiter is the largest planet.	5.
B.	Mars is the fourth planet from the sun	8.

The initial step to make a vocabulary of unique words and calculate TF for each document.

- Inverse Document Frequency (IDF).

It is the measure of the importance of a word. TF does not consider the importance of words. Some words such as 'of', and' can be more frequently present but are of little significance.

$$IDF(w, D) = \ln\left(\frac{\text{Total no. of docs. (N) in corpus D}}{\text{no. of docs containing w.}}\right)$$

Words	TF (for A)	TF (for B)	IDF.
Jupiter	1/5	0	$\ln(2/1) = 0.69$
is	1/5	1/8	$\ln(2/2) = 0$
the	1/5	2/8	$\ln(2/2) = 0.$
largest	1/5	0	$\ln(2/1) = 0.69$
planet	1/5	1/8	$\ln(2/2) = 0.$
mars	0	1/8	$\ln(2/1) = 0.69$
fourth	0	1/8	$\ln(2/1) = 0.69$
from	0	1/8	$\ln(2/1) = 0.69$
sun.	0	1/8.	$\ln(2/1) = 0.69.$

* TF-IDF.

- It is the product of TF and IDF.

TF IDF gives more weightage to the word that is rare in the corpus (all the documents).

TFIDF provides more importance to the word that is more frequent in the document.

$$TFIDF(w, d, D) = TF(w, d) * IDF(w, D).$$

4] Bag of Words (BoW).

- Machine learning algorithms cannot work with raw text directly. Rather, text must be converted into vectors of numbers.
- In natural language processing, a common technique for extracting features from text is to place all of the words that occur in the text in a bucket.
- This approach is called a bag of words model or BoW for short. It's refered to as a 'bag' of words because any information about the structure of the sentence is lost.

5] Algorithm for tokenization, pos tagging, stop words removal, stemming and lemmatization –

Step 1: Download the required packages.

```
nltk.download('punkt');
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')
```

Step 2: Initialize the text.

Step 3: Perform Tokenization.

```
# sentence tokenization.
from nltk.tokenize import sent-tokenize.
tokenized-text = sent-tokenize (text)
print (tokenized-text).

# word tokenization.
from nltk.tokenize. import word-tokenize.
tokenized-word = word-tokenize (text).
print (tokenized-word).
```

Step 4: Removing punctuations and stop words.

```
# print stop words of English.
from nltk.corpus import stopwords.
stop-words = set (stopwords.words ("english"))
print (stop-words).
```

Step 5: Perform Stemming.

```
from nltk.stem import PorterStemmer.
e-words = ["wait", "waiting", "waited", "waits"]
ps = PorterStemmer ().
for w in e-words:
        root Word = ps.stem (w)
print (root Word).
```

Step 6: Perform Lemmatization.

```
from nltk.stem import WordNetLemmatizer
wordnet-lemmatizer = WordNetLemmatizer ().
text = "studies studying cries cry".
tokenization = nltk.word-tokenize (text)
```

for w in tokenization :
print("lemma for () is {}".
format (w, wordnet_lemmatizer.lemmatize(w)))

Step 7: Apply POS tagging to text.

for word in words :
print("nltk.postag([word]))

6] Algorithm for create representation of document by calculating TFIDF. –

Step 1: Import the necessary libraries. - pandas and sklearn.

Step 2: Initialize the documents -
documentA = "Jupiter is the largest planet".
documentB = "Mars is the fourth planet from the sun".

Step 3: Create Bag of words for docA and docB.
bagofWordsA = documentA.split(' ')
bagofWordsB = documentB.split(' ').

Step 4: Create collection of unique words from document A and B.

uniqueWords = set(bagofWordsA).union(set(bagofWordsB).

Step 5: Create a dictionary of words and their occurence for each document in the corpus.

```
numofwordsA = dict.fromkeys(uniqueWords, 0).

for word in bagofWordsA:
        numofwordsA[word] += 1
numofwordsB = dict.fromkeys(uniqueWords, 0).
```

Step 6: Compute term frequency for each word.

```
tfA = computeTF(numofwordsA, bagofwordsA)
tfB = computeTF(numofwordsB, bag of wordsB).
```

Step 7: Compute the term inverse document frequency.

```
idfs = computeIDF([numofWordsA, numofwordsB])
```

Step 8: Compute TFIDF for all words.

```
tfidf = computeTFIDF(tfA, idfs)
tfidf = computeTFIDF(tfB, idfs).

df = pd.dataframe([tfidfA, tfidfB])

df.
```

Conclusion:

In this way we have done text data analysis using TF IDF algorithm.

* * * * * * *

Page No.
Date

* Differences between Stemming and lemmatization -

Stemming.	Lemmatization
1. Stemming is a process that stems or removes last few characters from a word, often leading to incorrect meanings and spelling.	1. Lemmatization considers the context and converts the word to its meaningful base form which is called lemma.
2. For instance, stemming the word Caring would return 'car'.	2. For instance lemmatizing the word 'caring' would return 'care'.
3. Stemming is used in case of large dataset where performance is an issue.	3. Lemmatization is computationally expensive since it involves look up tables and what not.