Assignment 2

Problem Statement:

Implement a program for retrieval of documents using inverted files.

Objective:

1. Evaluate and analyse retrieved information
2. To study Indexing, Inverted Files and searching with the help of inverted file

Theory:

An inverted index is an index data structure storing a mapping from content, such as words or numbers, to its locations in a document or a set of documents. In simple words, it is a HashMap like data structure that directs you from a word to a document or a web page.

Creating Inverted Index

We will create a **Word level inverted index** that is it will return the list of lines in which the word is present. We will also create a dictionary in which key values represent the words present in the file and the value of a dictionary will be represented by the list containing line numbers in which they are present. To create a file in Jupiter notebook, use magic function:

%%writefile file.txt

This is the first word.

This is the second text, Hello! How are you?

This is the third, this is it now.

This will create a file named file.txt will the following content.

To read file:

- Python3

```
# this will open the file

file = open('file.txt', encoding='utf8')

read = file.read()

file.seek(0)

read
```

```
 # to obtain the

# number of lines

# in file

line = 1

for word in read:

    if word == '\n':

        line += 1

print("Number of lines in file is: ", line)

 # create a list to

# store each line as

# an element of list

array = []

for i in range(line):

    array.append(file.readline())

 array
```

```
Number of lines in file is: 3

['This is the first word.\n',

'This is the second text, Hello! How are you?\n',

'This is the third, this is it now.']
```

Functions used:

- **Open:** It is used to open the file.
- **read:** This function is used to read the content of the file.
- **seek(0):** It returns the cursor to the beginning of the file.

Remove punctuation:

Python3

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

for ele in read:
```

```
    if ele in punc:
        read = read.replace(ele, " ")
  read
 # to maintain uniformity
read=read.lower()
read
```

Output:

'this is the first word \n

this is the second text hello how are you \n

this is the third this is it now '

Tokenize the data as individual words:

Apply linguistic preprocessing by converting each word in the sentences into tokens. Tokenizing the sentences help with creating the terms for the upcoming indexing operation.

Python3

```
def tokenize_words(file_contents):
    """
    Tokenizes the file contents.
     Parameters
    ----------
    file_contents : list
        A list of strings containing the contents of the file.
        Returns
    -------
    list
        A list of strings containing the contents of the file tokenized.

    """
```

```
    result = []
     for i in range(len(file_contents)):
        tokenized = []
         # print("The row is ", file_contents[i])
         # split the line by spaces
        tokenized = file_contents[i].split()
         result.append(tokenized)
     return result
```

Clean data by removing stopwords:

Stop words are those words that have no emotions associated with it and can safely be ignored without sacrificing the meaning of the sentence.

Python3

```
from nltk.tokenize import word_tokenize
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
 for i in range(1):
    # this will convert
    # the word into tokens
    text_tokens = word_tokenize(read)
 tokens_without_sw = [
    word for word in text_tokens if not word in stopwords.words()]
 print(tokens_without_sw)
```

Output:

['first', 'word', 'second', 'text', 'hello', 'third']

Create an inverted index:

Python3

```
dict = {}
 for i in range(line):
    check = array[i].lower()
    for item in tokens_without_sw:
        if item in check:
            if item not in dict:
                dict[item] = []
            if item in dict:
                dict[item].append(i+1)
 dict
```

Output:

{'first': [1],

'word': [1],

'second': [2],

'text': [2],

'hello': [2],

'third': [3]}

Conclusion:

By this way, we can perform retrieval of documents using inverted files.

Oral Questions?

1. What is mean by inverted index?
2. What are steps for creation of Inverted Index?
3. What are built in functions used for index creation?

Assignment 3