**CSE-3024 Web Mining**

**Digital Assignment 1**

**Alokam Nikhitha**

**19BCE2555**

**CSE-3024 Web Mining**

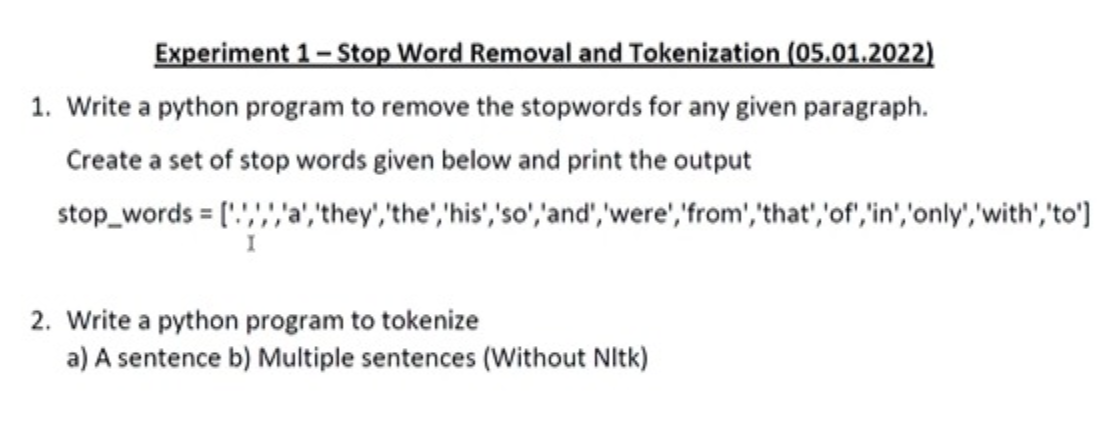
**Lab Assignment 1**

**Alokam Nikhitha**

**19BCE2555**

Question 1

**Problem statement:**



**Procedure:**

* At First, we import the text file in our work space. To do this we can use open method of python which reads the file into our workspace.
* Next, we read each word into a variable as string using a nested for loop wherein we split each word whenever we encounter a space.
* Next, using regex in python we remove the punctuations from our string input. This will make sure that tokens are free from sentence structure.
* We tokenize each token in our text using split() function of NumPy lists and save it in a list.
* Then we use NumPy’s unique method to only include unique tokens from our identified set of tokens.
* A tentative list of stop words and using a nested for loop we check if the given token belongs to that list or not, to remove stopwords. If it doesn’t then we save it else we discard it.
* Finally, we print our list that contains the resultant tokens after removal of stop words.

**Code:**

#Reading input from a text file and saving it as a string

text = ""

with open('test\_file.txt') as file:

for line in file:

for word in line.split():

text= text + " " + word

#Removing punctuations from our input file

import re

text = re.sub(r'[^\w\s]', '', text)

text

#Printing each token

print(text.split())

#Printing unique tokens

import numpy as np

print(np.unique(text.split()))

#Removing StopWords

stopwords = ["i", "a", "am", "and", "at", "for", "in", "is", "my", "of", "this"]

res = []

for x in tokens:

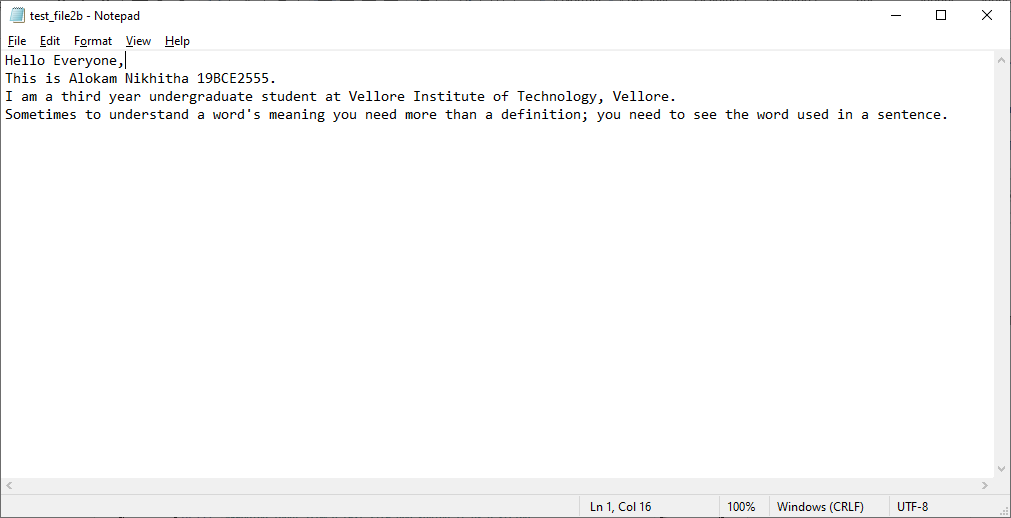
if x not in stopwords:

res.append(x)

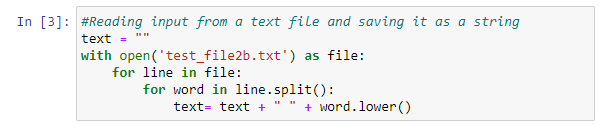
#Printing cleaned tokens in our input text file

print(res)

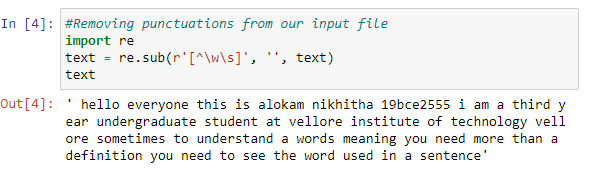
**Text File Taken as Input:**

****

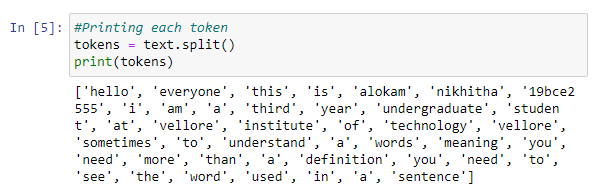
**Code Snippets and Outputs:**



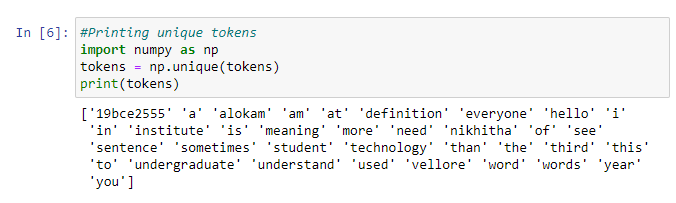
Here we are reading the text file using open method in python. Then reading each line we split each word and append it to a string variable with a space in between.



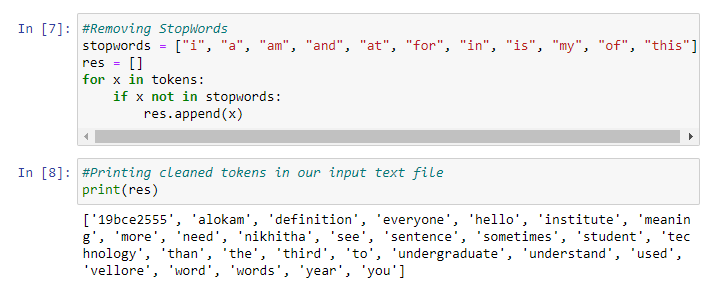
Here we are removing punctuations from our input file. This is done using regex, where we keep only alphanumeric inputs in our text string. We can see all the periods and commas from original input files are removed here.



Next, we are splitting each word in our string using space character. Clearly, they form a token and hence we print each token.



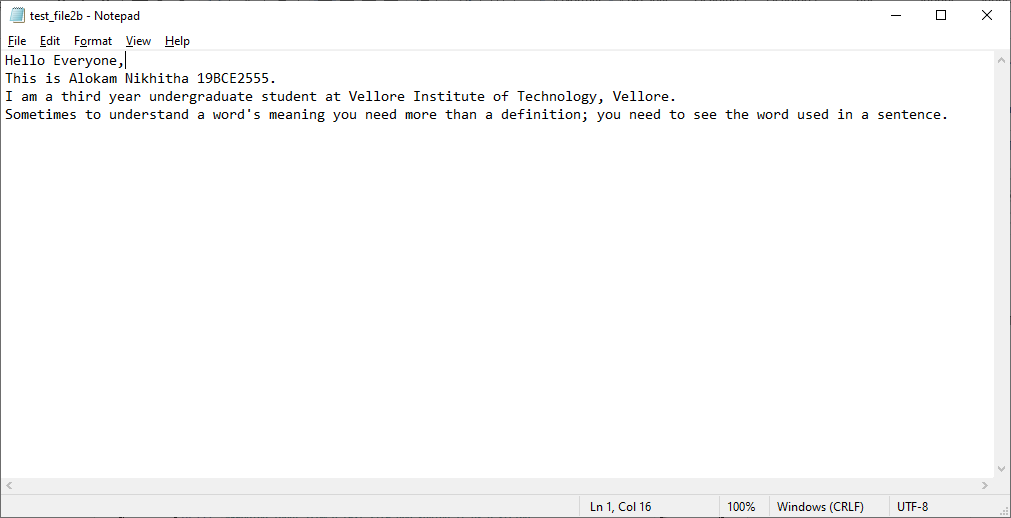
Here we are only printing unique tokens from all the generated tokens using split method. This is done using NumPy’s unique function, which identifies all the unique elements from a list.



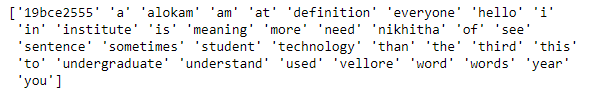
Here we remove all the stop words from a self-defined list of stop words. We use nested loop to check if given token belongs to both tokens list and stopwords list. If it does, we don’t add it to our result else we add it to our results.

**Results and Output**

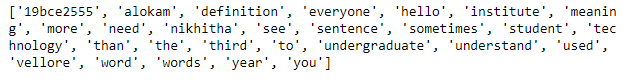
**- Input text:**

****

**- Tokens of input text:**

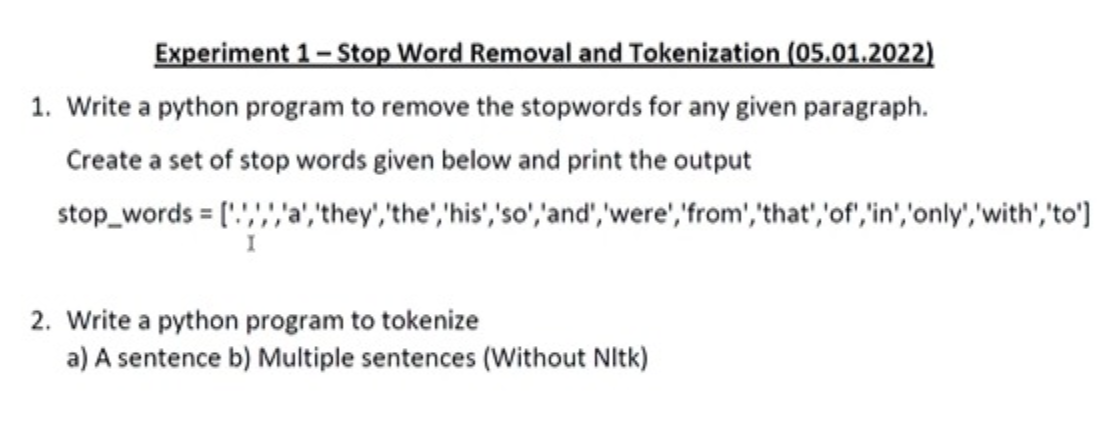
****

**-Tokens after removal of stop words:**

****

Question 2

**Problem statement:**



**Procedure:**

* Firstly, we import the text file in our work space. To do this we can use open method of python which reads the file into our workspace.
* Next, we read each word into a variable as string. This can be done using a nested for loop wherein we split each word whenever we encounter a space.
* Next, using regex in python we remove the punctuations from our string input. This will make sure that tokens are free from sentence structure.
* Finally, we will print each token and then unique tokens.

**a)**

**Code:**

#Reading input from a text file and saving it as a string

text = ""

with open('test\_file2a.txt') as file:

for line in file:

for word in line.split():

text= text + " " + word

#Removing punctuations from our input file

import re

text = re.sub(r'[^\w\s]', '', text)

text

#Printing each token

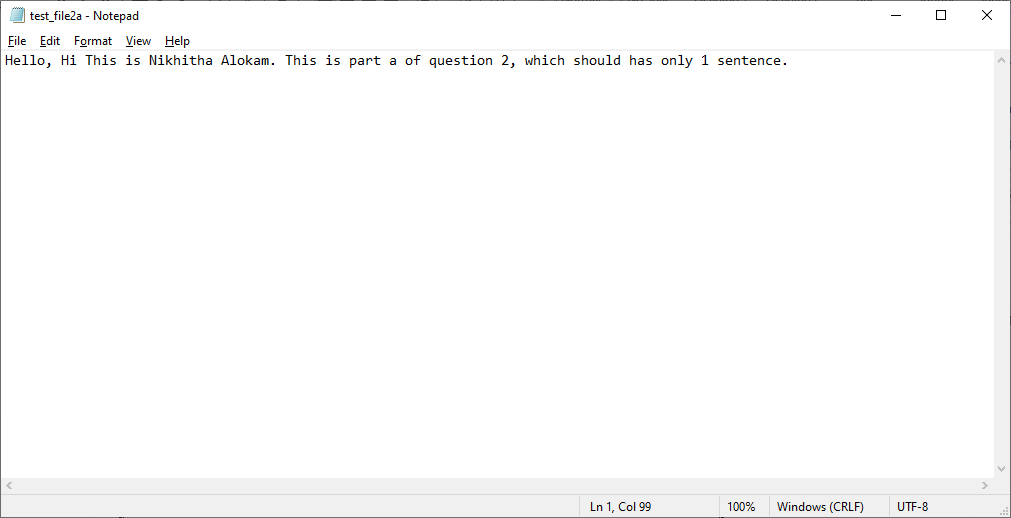
print(text.split())

#Printing unique tokens

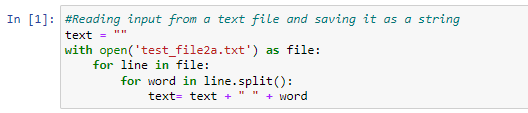
import numpy as np

print(np.unique(text.split()))

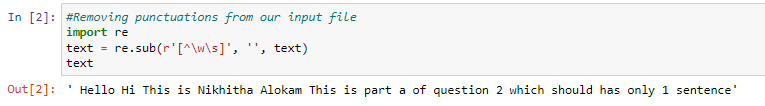
**Text File Taken as Input:**

****

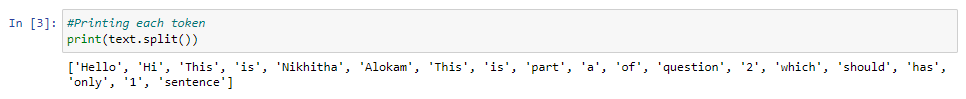
**Code Snippets and Outputs:**



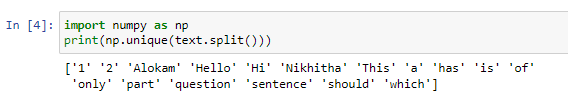
Here we are reading the text file using open method in python. Later we are reading each line we split each word and append it to a string variable with a space in between.



Here we are removing punctuations from our input file using regex, where we keep only alphanumeric inputs in our text string. We can see all the periods and commas from original input files are removed here.



Next, we are splitting each word in our string using space character. Clearly, they form a token and hence we print each token.



Here we are only printing unique tokens from all the generated tokens using split method. This is done using NumPy’s unique function, which identifies all the unique elements from a list.

**Results and Output**

**- Input text:**

****

**- Tokens of input text:**

****

**b)**

**Code:**

#Reading input from a text file and saving it as a string

text = ""

with open('test\_file2b.txt') as file:

for line in file:

for word in line.split():

text= text + " " + word

#Removing punctuations from our input file

import re

text = re.sub(r'[^\w\s]', '', text)

text

#Printing each token

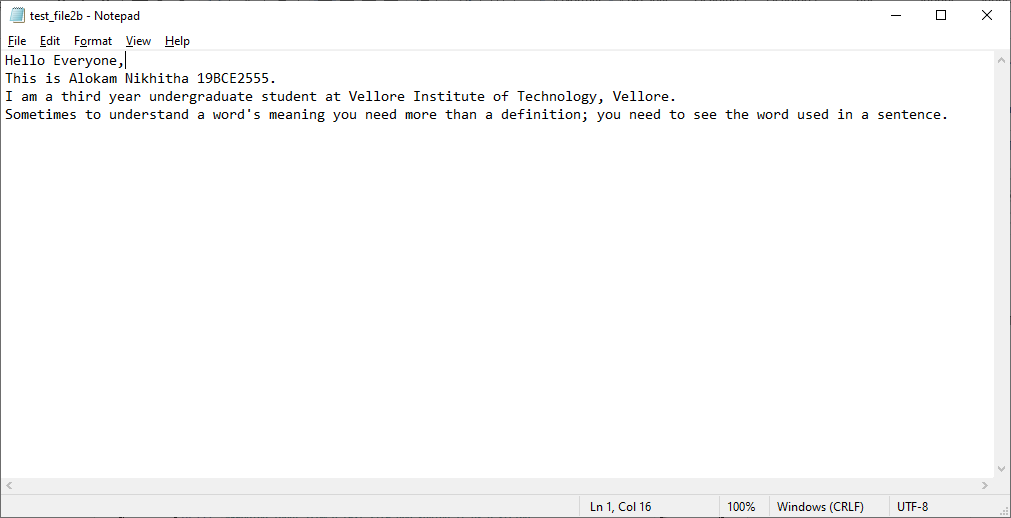
print(text.split())

#Printing unique tokens

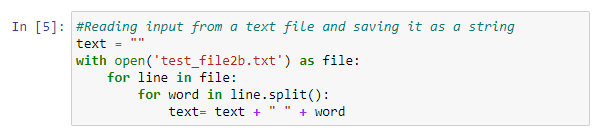
import numpy as np

print(np.unique(text.split()))

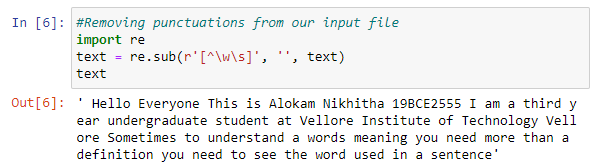
**Text File Taken as Input:**

****

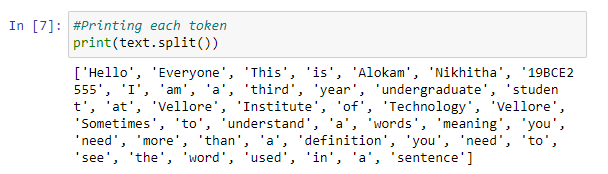
**Code Snippets and Outputs:**



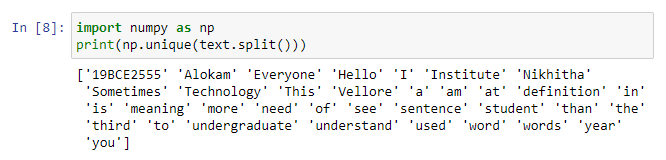
Here we are reading the text file using open method in python. Then reading each line we split each word and append it to a string variable with a space in between.



Here we are removing punctuations from our input file. This is done using regex, where we keep only alphanumeric inputs in our text string. We can see all the periods and commas from original input files are removed here.



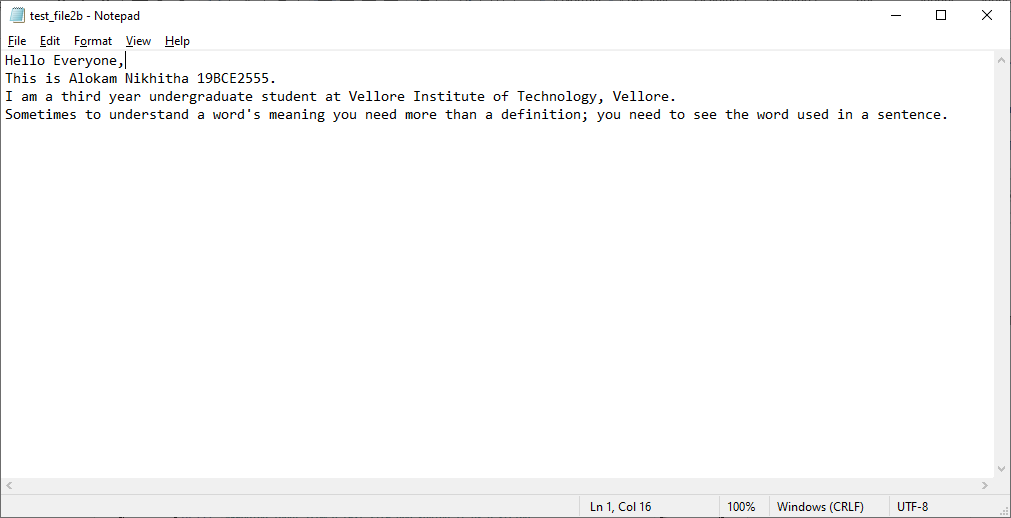
Next, we are splitting each word in our string using space character. Clearly, they form a token and hence we print each token.



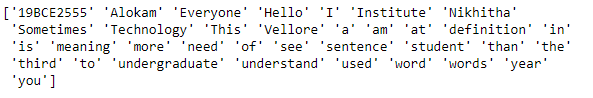
Here we are only printing unique tokens from all the generated tokens using split method. This is done using NumPy’s unique function, which identifies all the unique elements from a list.

**Results and Output:**

**- Input text:**

****

**- Tokens of input text:**

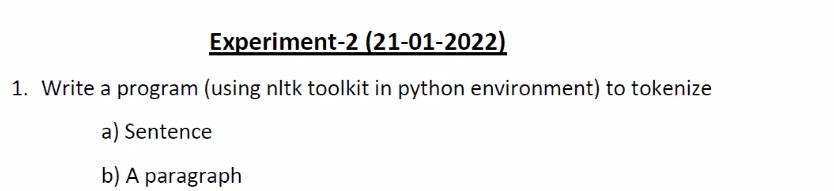
****

**CSE-3024 Web Mining**

**Lab Assignment 2**

**Alokam Nikhitha**

**19BCE2555**

Question 

**Problem statement:**

To write a program to tokenize using nltk toolkit in Python Environment.

**Procedure:**

* We firstly import our text file into our workspace. To do that we are able to use open method of python which will read our text file to the workspace.
* Next, we will import the necessary NLTK libraries including stopwords, sent\_tokenize and word\_tokenize.
* Using word\_tokenize we tokenize each word and store it in a variable list named tokens
* Then, we will read every word in input file as a string input. This may be carried out using nested for loop in which we split every word whenever we come across a space.
* Then , we use regex to remove the punctuations from the input string. This will render token a higher syntactic shape and break the sentence bonds.
* Then ,we split and store every token right into a list with nltk’s sentence\_tokenize method.
* Then subsequently we remove stop words in the same process as in preceding assignment.
* Finally, we print our list that contains the resultant tokens post removal of stop words as well.

**a) Sentence**

**Code:**

#Reading input (Single ) from a text file

text = ""

with open('test\_file2a.txt') as file:

for line in file:

for word in line.split():

text = text + " " + word.lower()

#Importing libraries

import re

import nltk

from nltk.tokenize import sent\_tokenize, word\_tokenize

from nltk.corpus import stopwords

#Removing punctuations from our input .

text = re.sub(r'[^\w\s]', '', text)

text

#Printing each token

tokens = word\_tokenize(text)

print(tokens)

#Printing unique tokens

import numpy as np

tokens = np.unique(tokens)

print(tokens)

#Removing Stopwords

res = []

for x in tokens:

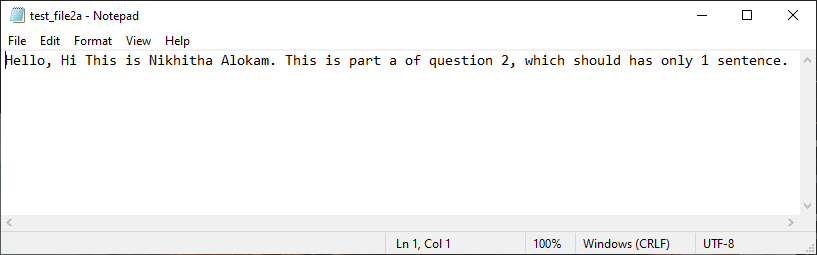
if x not in set(stopwords.words('english')):

res.append(x)

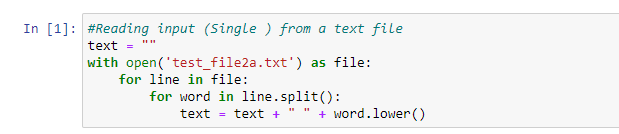
#Printing cleaned tokens in our input text file

print(res)

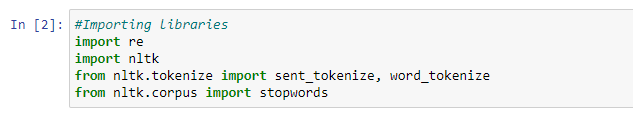
**Text File Taken as Input:**

****

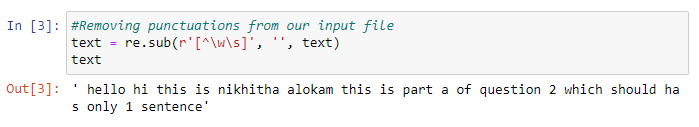
**Code Snippets and Outputs:**

****

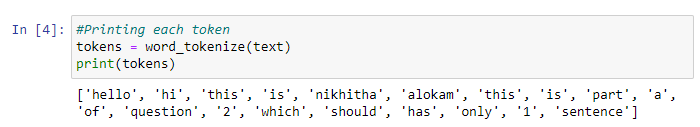
Here we're analyzing the text file by using open approach in python. Then analyzing every line we split every word and append it to a string variable with a space in between.



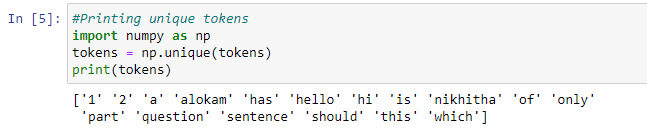
Here we're importing the necessary libraries which incorporates our library of challenge this is nltk. We additionally import stopwords and word\_tokenize, sentence\_tokenize from nltk. To eliminate punctuations we import the regex library.



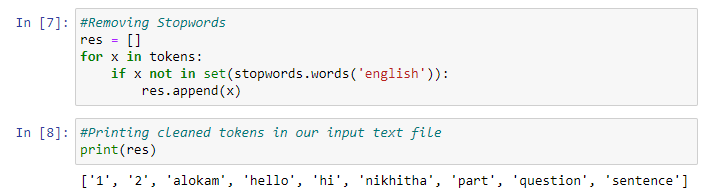
Here we're removing punctuations from the input taken from input file. This is carried out by using regex, wherein we keep only alphanumeric inputs in our text string. We can see all of the intervals and commas from original input file are eliminated here.



Since we have to tokenize every word, we have used the word\_tokenize and as we can see each token in identified here and we print them.



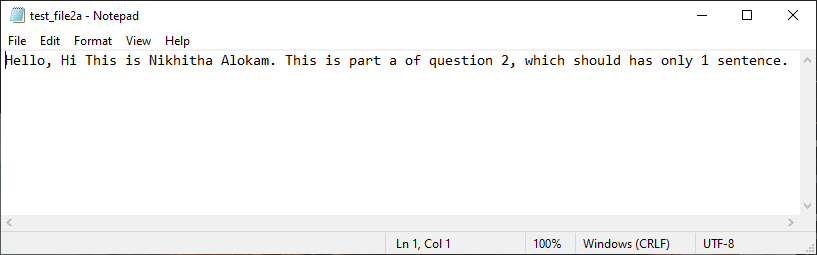
Here we print all the unique tokens in after tokenizing. And print them.



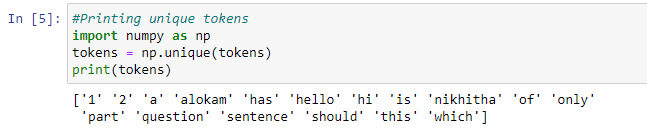
Here we remove all the stop words from a self-defined list of stop words. We use nested loop to check if given token belongs to both tokens list and stopwords list. If it does, we don’t add it to our result else we add it to our results.

**Results and Output**

1. Input Sentence

****

1. Tokens with out removing Stopwords.



1. Tokens after removing Stopwords



**b) A Paragraph**

**Code:**

#Reading input (Single ) from a text file

text = ""

with open('test\_file2b.txt') as file:

for line in file:

for word in line.split():

text = text + " " + word.lower()

#Importing libraries

import re

import nltk

from nltk.tokenize import sent\_tokenize, word\_tokenize

from nltk.corpus import stopwords

#Removing punctuations from our input .

text = re.sub(r'[^\w\s]', '', text)

text

#Printing each token

tokens = word\_tokenize(text)

print(tokens)

#Printing unique tokens

import numpy as np

tokens = np.unique(tokens)

print(tokens)

#Removing Stopwords

res = []

for x in tokens:

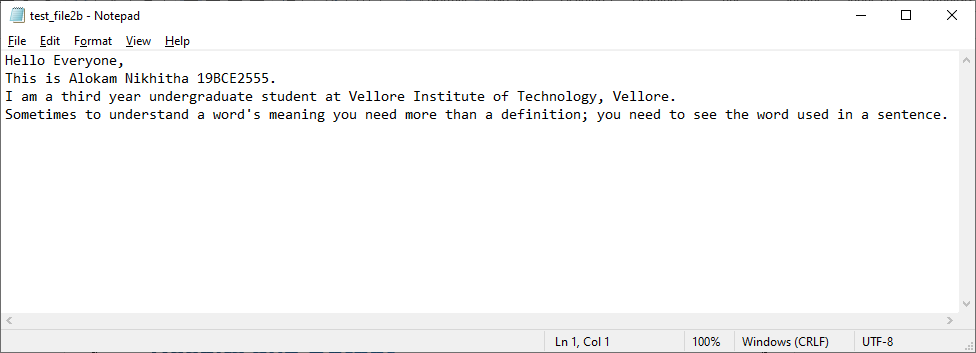
if x not in set(stopwords.words('english')):

res.append(x)

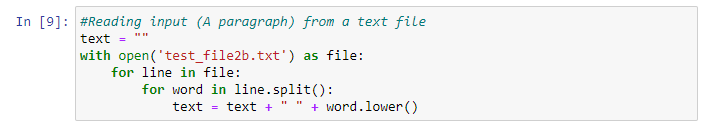
#Printing cleaned tokens in our input text file

print(res)

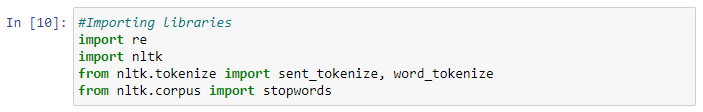
**Text File Taken as Input:**

****

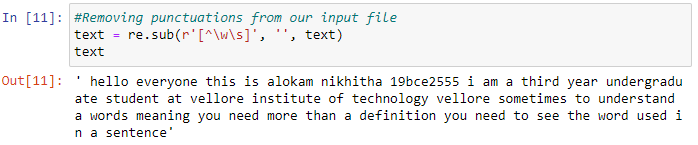
**Code Snippets and Outputs:**

****

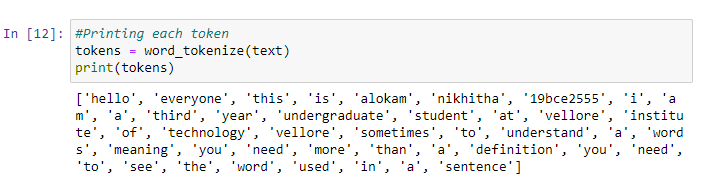
Here we're analyzing the text file by using open approach in python. Then analyzing every line we split every word and append it to a string variable with a space in between.



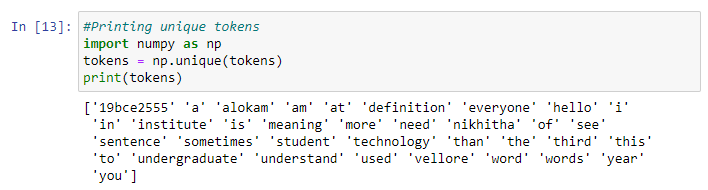
Here we're importing the necessary libraries which incorporates our library of challenge this is nltk. We additionally import stopwords and word\_tokenize, sentence\_tokenize from nltk. To eliminate punctuations we import the regex library



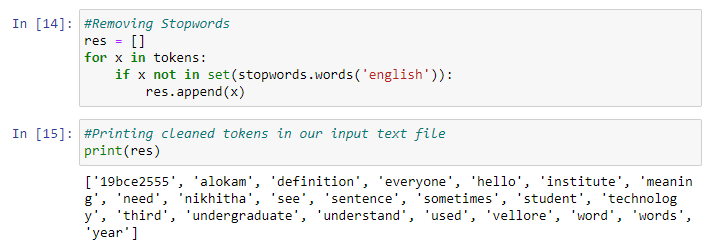
Here we're removing punctuations from the input taken from input file. This is carried out by using regex, wherein we keep only alphanumeric inputs in our text string. We can see all of the intervals and commas from original input file are eliminated here.



Since we have to tokenize every word, we have used the word\_tokenize and as we can see each token in identified here and we print them.



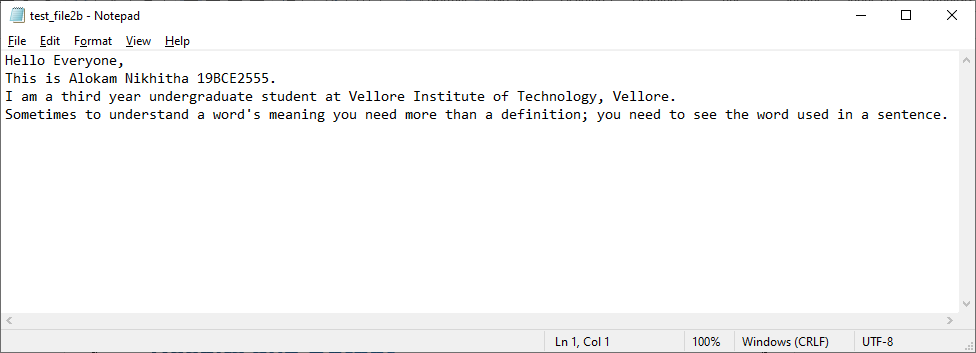
Here we print all the unique tokens in after tokenizing. And print them.

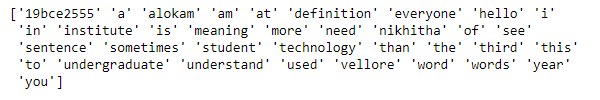


Here we remove all the stop words from a self-defined list of stop words. We use nested loop to check if given token belongs to both tokens list and stopwords list. If it does, we don’t add it to our result else we add it to our results.

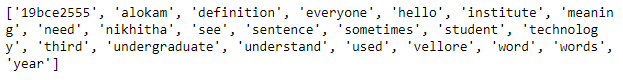
**Results and Output**

1.Input Sentence

****2.Tokens with out removing Stopwords.

****

3.Tokens after removing Stopwords



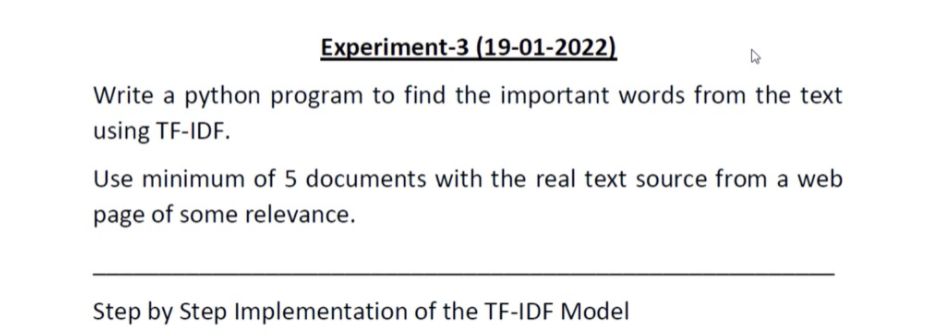
**CSE-3024 Web Mining**

**Lab Assignment 3**

**Alokam Nikhitha**

**19BCE2555**

Question



**Problem statement:**

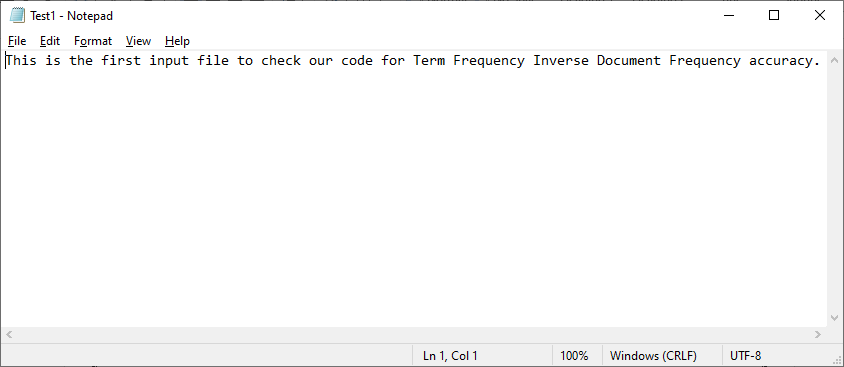
Python program to find the important words from the textfile using TF-IDF using atleast minimum of 5 documents

**Procedure:**

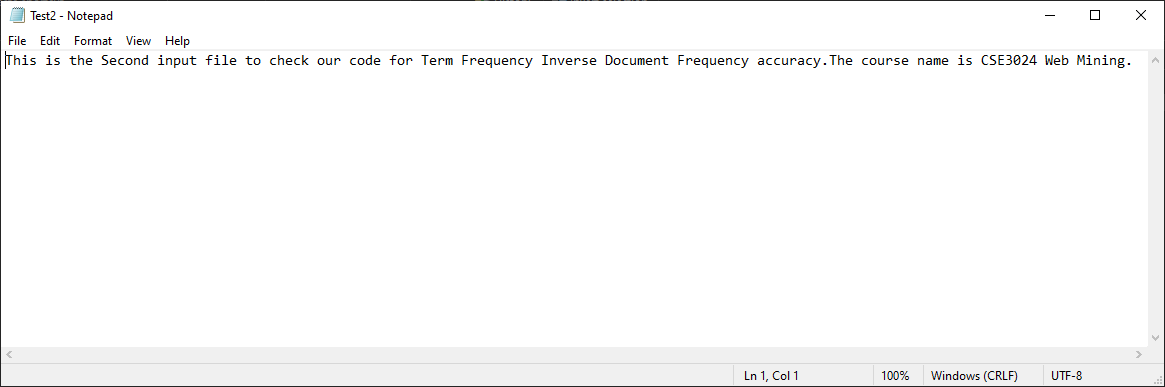
* We will Firstly import our libraries Which are required in doing the term frequency count.
* Later, we will declare and define tf, idf, n\_containing and tf\_idf functions that will help assist the return values and make code more readable.
* We will create 5 Text File inputs and read them in our workspace.
* Later, We will make the bloblist that contains all the Text File Inputs in list format. And then we will print the counts of top 3 words in every document.
* We will then calculate the cosine similarity using inbuilt cosine\_similarity matrix.
* For the above we need to create a pandas data frame of count vectors.

**Text File Taken as Input:**

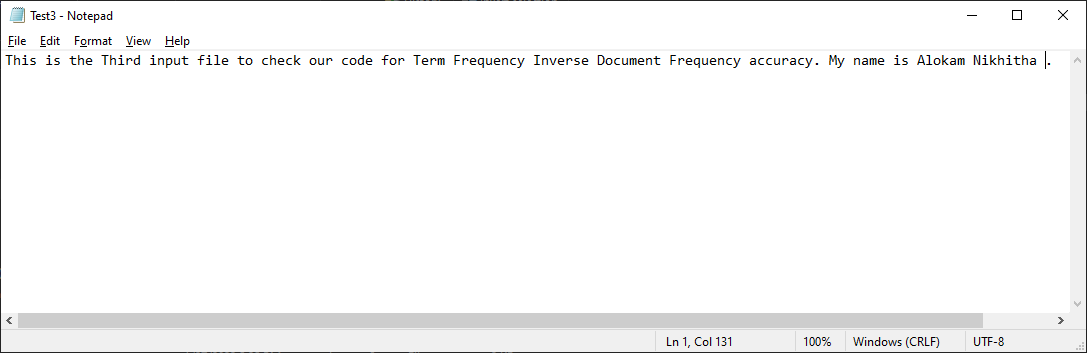
**Text File 1:**

****

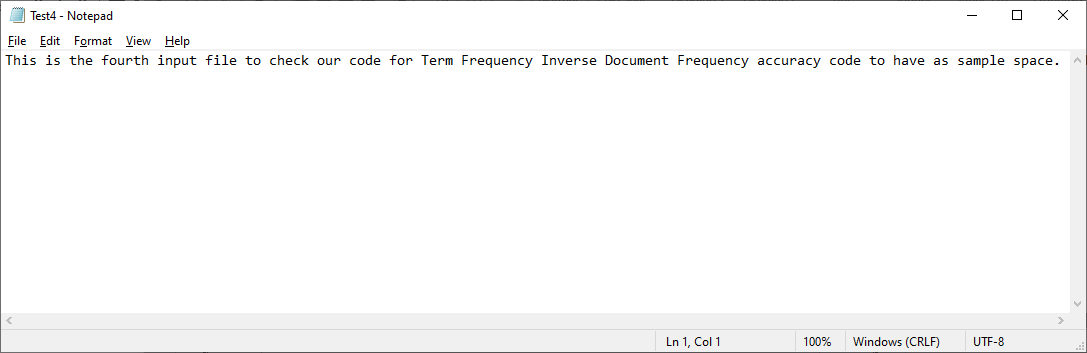
**Text File 2:**

****

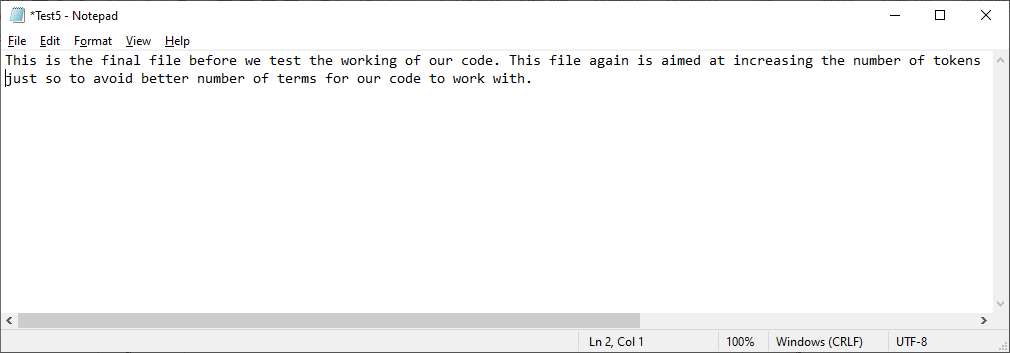
**Text File 3:**



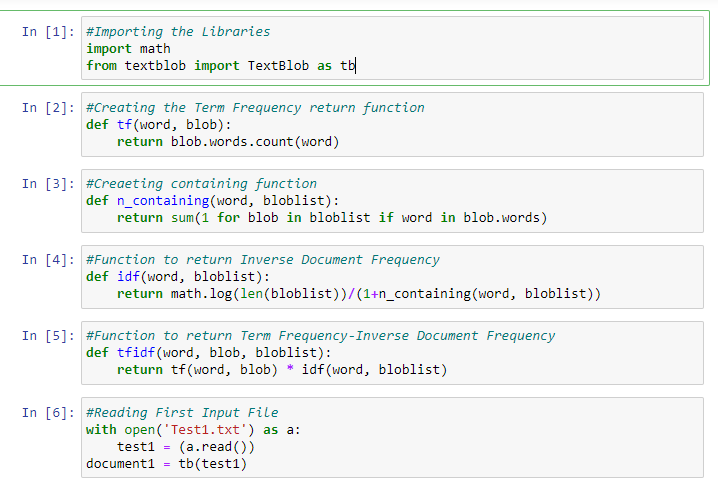
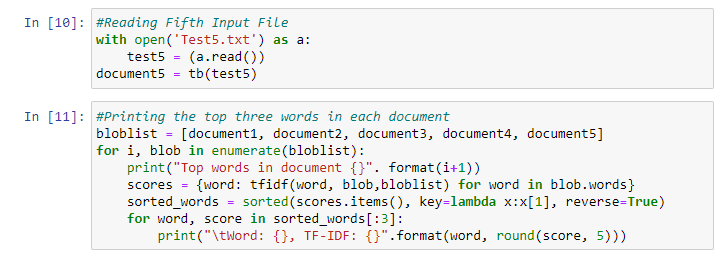
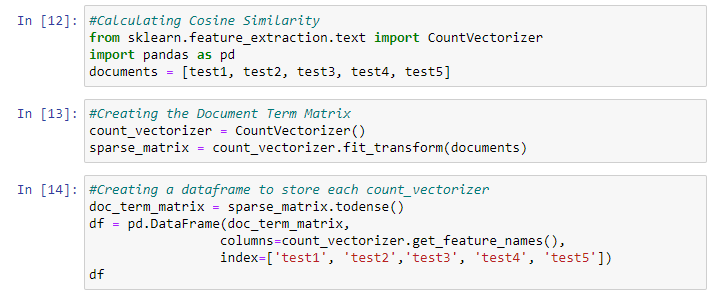
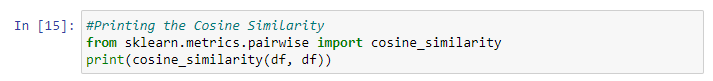
**Text File 4:**



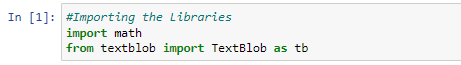
**Text File 5:**



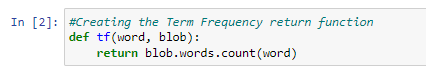
**Code:**

 ****   

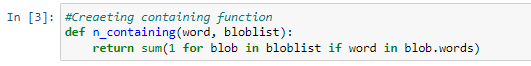
**Code Snippets and Outputs:**



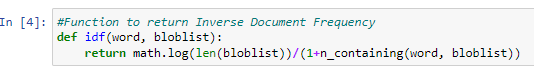
Here we are importi8ng the necessary Libraries



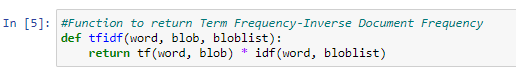
Here we are creating the Term Frequency return Function which takes word and blob as attributes.



Here we are now creating the n\_containing Function wwhich takes words and bloblist as attributes.



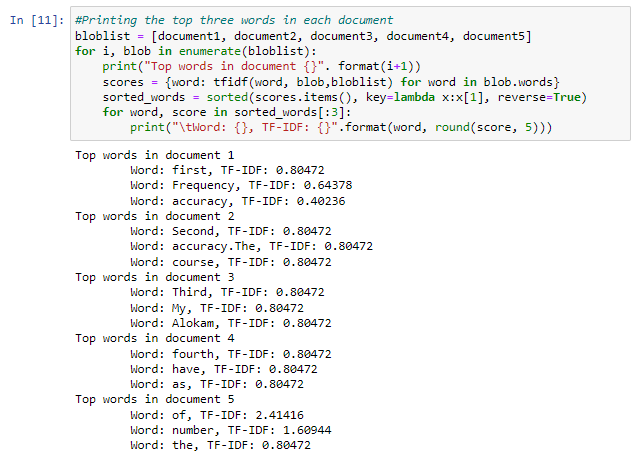
A function named idf is created inorder to Inverse the Document Frequency



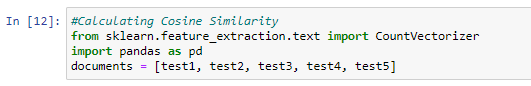
Here we create a Function named ifidtf to return Term Frequency-Inverse Document Frequency



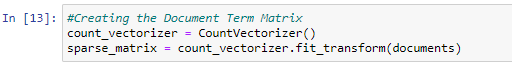
Here we are reading all the 5 Input Text Files(i.e, Test1.txt, Test2.txt, Test3.txt, Test4.txt, Test5.txt)

****

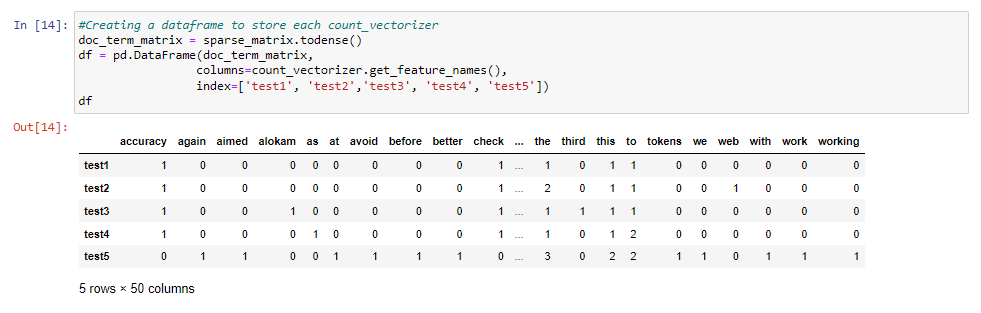
Here we’ve printed the top words in every document. We’ve printed only top 3 words and the TF-IDF values of them in the same line with the word/term.



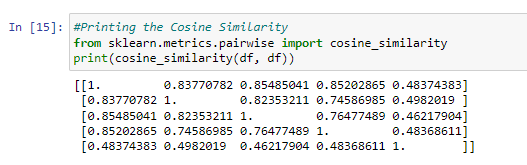
Here we are Calculating the Cosine Similarity of all the Input Text files.



Here we’ve created count vector which contains the frequency of each word of each document. This is for finding the Cosine Similarity.



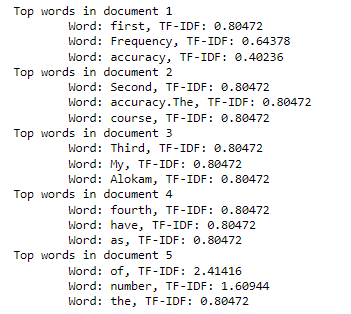
Here we’ve combined the count vectors of every document into Pandas Data Frame.



Here we printed Csonne Similarity of Every Document

**Results and Output**

Top words in Each Input Text file:

****

Cosine similarity

