+ Code + Text

Name: R SHRI VAIBBAV

Reg No: 20BCE1495

Course Code: CSE4022 Natural Language Processing

Faculty: Dr. V. Maria Anu

1. Utilize Python NLTK (Natural Language Tool Kit) Platform and do the following. Install relevant Packages and Libraries

```
pip install nltk
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: nltk in /usr/local/lib/python3.8/dist-packages (3.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.8/dist-packages (from nltk) (1.2.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from nltk) (4.64.1)
Requirement already satisfied: click in /usr/local/lib/python3.8/dist-packages (from nltk) (7.1.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.8/dist-packages (from nltk) (2022.6.2)
```

```
import nltk
nltk.download("popular")
```

```
[nltk_data] Downloading collection 'popular'
[nltk_data]    |
[nltk_data]    | Downloading package cmudict to /root/nltk_data...
[nltk_data]    |   Package cmudict is already up-to-date!
[nltk_data]    | Downloading package gazetteers to /root/nltk_data...
[nltk_data]    |   Package gazetteers is already up-to-date!
[nltk_data]    | Downloading package genesis to /root/nltk_data...
[nltk_data]    |   Package genesis is already up-to-date!
[nltk_data]    | Downloading package gutenberg to /root/nltk_data...
[nltk_data]    |   Package gutenberg is already up-to-date!
[nltk_data]    | Downloading package inaugural to /root/nltk_data...
[nltk_data]    |   Package inaugural is already up-to-date!
[nltk_data]    | Downloading package movie_reviews to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package movie_reviews is already up-to-date!
[nltk_data]    | Downloading package names to /root/nltk_data...
[nltk_data]    |   Package names is already up-to-date!
[nltk_data]    | Downloading package shakespeare to /root/nltk_data...
[nltk_data]    |   Package shakespeare is already up-to-date!
[nltk_data]    | Downloading package stopwords to /root/nltk_data...
[nltk_data]    |   Package stopwords is already up-to-date!
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Package treebank is already up-to-date!
[nltk_data]    | Downloading package twitter_samples to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package twitter_samples is already up-to-date!
[nltk_data]    | Downloading package omw to /root/nltk_data...
[nltk_data]    |   Package omw is already up-to-date!
[nltk_data]    | Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]    |   Package omw-1.4 is already up-to-date!
[nltk_data]    | Downloading package wordnet to /root/nltk_data...
[nltk_data]    |   Package wordnet is already up-to-date!
[nltk_data]    | Downloading package wordnet2021 to /root/nltk_data...
[nltk_data]    |   Package wordnet2021 is already up-to-date!
[nltk_data]    | Downloading package wordnet31 to /root/nltk_data...
[nltk_data]    |   Package wordnet31 is already up-to-date!
[nltk_data]    | Downloading package wordnet_ic to /root/nltk_data...
[nltk_data]    |   Package wordnet_ic is already up-to-date!
[nltk_data]    | Downloading package words to /root/nltk_data...
[nltk_data]    |   Package words is already up-to-date!
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package maxent_ne_chunker is already up-to-date!
[nltk_data]    | Downloading package punkt to /root/nltk_data...
[nltk_data]    |   Package punkt is already up-to-date!
[nltk_data]    | Downloading package snowball_data to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package snowball_data is already up-to-date!
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Package averaged_perceptron_tagger is already up-
[nltk_data]    |       to-date!
[nltk_data]    |
[nltk_data]  Done downloading collection popular
True
```

```
from nltk.tokenize import word_tokenize
from nltk.corpus import brown
from nltk.tokenize import sent_tokenize
from collections import defaultdict
from urllib import request
from collections import Counter
```

```
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer
```

```
", ".join(brown.raw ())
```

```
'\n, \n, \t, T, h, e, /, a, t,  , F, u, l, t, o, n, /, n, p, -, t, l,  ,
C, o, u, n, t, y, /, n, n, -, t, l,  , G, r, a, n, d, /, j, j, -, t, l,
, J, u, r, y, /, n, n, -, t, l,  , s, a, i, d, /, v, b, d,  , F, r, i, d,
a, y, /, n, r,  , a, n, /, a, t,  , i, n, v, e, s, t, i, g, a, t, i, o,
n, /, n, n,  , o, f, /, i, n,  , A, t, l, a, n, t, a, ', s, /, n, p, $,
, r, e, c, e, n, t, /, j, j,  , p, r, i, m, a, r, y, /, n, n,  , e, l, e,
c, t, i, o, n, /, n, n,  , p, r, o, d, u, c, e, d, /, v, b, d,  , `, `,
/, `, `,  , n, o, /, a, t,  , e, v, i, d, e, n, c, e, /, n, n,  , ', ',
```

```
text = brown.words()
data = ' '.join([str(elem) for elem in text])
data
```

```
'The Fulton County Grand Jury said Friday an investigation of Atlanta's r
ecent primary election produced `` no evidence '' that any irregularities
took place . The jury further said in term-end presentments that the City
Executive Committee , which had over-all charge of the election , `` dese
rves the praise and thanks of the City of Atlanta '' for the manner in wh
ich the election was conducted . The September-October term jury had been
charged by Fulton Superior Court Judge Durwood Pye to investigate reports
of possible `` irregularities '' in the hard-fought primary which was won
```

```
#1.1-Explore Brown Corpus and find the size, tokens, categories
print("Explore Brown Corpus and find the size, tokens, categories")
print(brown.categories())
len(brown.categories())
```

```
Explore Brown Corpus and find the size, tokens, categories
['adventure', 'belles_lettres', 'editorial', 'fiction', 'government', 'hobbies', 'humor', 'learned', 'lore', 'mystery', 'news', 're
15
```

```
#1.2-Find the size of word tokens
print("Find the size of word tokens")
print(len(word_tokenize(data)))
```

```
Find the size of word tokens
1173755
```

```
#1.3-Find the size of the category “government”
print("Find the size of the category “government” ")
print(len(brown.words(categories='government')))
```

```
Find the size of the category “government”
70117
```

```
#1.4-List the most frequent tokens
from nltk.probability import FreqDist
print("Most Frequent tokens:")
fdist1 = FreqDist(data)
fdist1
```

```
Most Frequent tokens:
FreqDist({' ': 1161191, 'e': 589980, 't': 423392, 'a': 371418, 'o': 357020, 'i': 333212, 'n': 332908, 's': 300431, 'r': 287337,
'h': 249219, ...})
```

```
#1.5-Count the number of sentences
print("Number of sentences:")
number_of_sentences = sent_tokenize(data)
print(len(number_of_sentences))
```

```
Number of sentences:
53190
```

2. Explore the corpora available in NLTK (any two)

- Raw corpus
- POS tagged
- Parsed
- Multilingual aligned
- Spoken language
- Semantic tagged

Raw Corpus

```
#Accessing Text from the Web
from urllib import request
url = "http://www.gutenberg.org/files/2554/2554-0.txt"
response = request.urlopen(url)
raw = response.read().decode('utf8')
type(raw)
```

```
str
```

```
len(raw)
```

```
1176812
```

```
raw[:75]
```

```
'\ufeffThe Project Gutenberg eBook of Crime and Punishment, by Fyodor Dos
```

```
#Strings- Text Processing at the Lowest Level

string = "Hi my name is vaibbav and i am a cse student in VIT chennai."
#Accessing Individual Characters
print(string[11])
```

```
i
```

```
#Accessing Substrings
print(string[-8:-1])
print( string[6:21])
```

```
chennai
name is vaibbav
```

```
# string split by whitespaces
print("\nConverted String:")
print(string.split())
```

```
Converted String:
['Hi', 'my', 'name', 'is', 'vaibbav', 'and', 'i', 'am', 'a', 'cse', 'student', 'in', 'VIT', 'chennai.']
```

```
# string to upper case
print("\nConverted String:")
print(string.upper())
```

```
Converted String:
HI MY NAME IS VAIBBAV AND I AM A CSE STUDENT IN VIT CHENNAI.
```

```
#string to lower case
print("\nConverted String:")
print(string.lower())
```

```
Converted String:
hi my name is vaibbav and i am a cse student in vit chennai.
```

```
#Extracting encoded text from files
path = nltk.data.find("/content/sample.txt")
f = open(path, encoding='latin2')
for line in f:
    line = line.strip()
    print(line.encode('unicode_escape'))
```

```
b'Hi my name is vaibbav and i am a cse student in VIT chennai.'
```

Parsed Corpora

The Treebank corpora provide a syntactic parse for each sentence. The NLTK data package includes a 10% sample of the Penn Treebank (in treebank), as well as the Sinica Treebank (in sinica_treebank)

```
from nltk.corpus import treebank

#Reading the Penn Treebank (Wall Street Journal sample):
print(treebank.fileids())
```

```
['wsj_0001.mrg', 'wsj_0002.mrg', 'wsj_0003.mrg', 'wsj_0004.mrg', 'wsj_0005.mrg', 'wsj_0006.mrg', 'wsj_0007.mrg', 'wsj_0008.mrg', 'w
```

```
print(treebank.words('wsj_0003.mrg'))
```

```
['A', 'form', 'of', 'asbestos', 'once', 'used', '*', ...]
```

```
print(treebank.tagged_words('wsj_0003.mrg'))
```

```
[('A', 'DT'), ('form', 'NN'), ('of', 'IN'), ...]
```

```
print(treebank.parsed_sents('wsj_0003.mrg')[1])
```

```
(S
  (S-TPC-2
    (NP-SBJ
      (NP (DT The) (NN asbestos) (NN fiber))
      (, ,)
      (NP (NN crocidolite))
      (, ,))
    (VP
      (VBZ is)
      (ADJP-PRD (RB unusually) (JJ resilient))
      (SBAR-TMP
        (IN once)
        (S
          (NP-SBJ (PRP it))
          (VP (VBZ enters) (NP (DT the) (NNS lungs)))))
      (, ,)
      (PP
        (IN with)
        (S-NOM
          (NP-SBJ
            (NP (RB even) (JJ brief) (NNS exposures))
            (PP (TO to) (NP (PRP it))))
          (VP
            (VBG causing)
            (NP
              (NP (NNS symptoms))
              (SBAR
                (WHNP-1 (WDT that))
                (S
                  (NP-SBJ (-NONE- *T*-1))
                  (VP
                    (VBP show)
                    (PRT (RP up))
                    (ADVP-TMP (NP (NNS decades)) (JJ later)))))))))))
  (, ,)
  (NP-SBJ (NNS researchers))
  (VP (VBD said) (SBAR (-NONE- 0) (S (-NONE- *T*-2))))
  (. .))
```

3. Create a text corpus with a minimum of 200 words (unique content). Implement the following text processing

```
#Creating a text corpus

import os
import nltk.data
from nltk.corpus.reader.plaintext import PlaintextCorpusReader

corpusdir = '/content/Corpus.txt' # Directory of corpus.

newcorpus = PlaintextCorpusReader(corpusdir, '.*')

newdata=nltk.data.load(corpusdir)
print(newdata)
```

```
While natural language processing isn’t a new science, the technology is rapidly advancing thanks to an increased interest in humar
As a human, you may speak and write in English, Spanish or Chinese.
But a computer’s native language – known as machine code or machine language – is largely incomprehensible to most people.
At your device’s lowest levels, communication occurs not with words but through millions of zeros and ones that produce logical act
Indeed, programmers used punch cards to communicate with the first computers 70 years ago.
This manual and arduous process was understood by a relatively small number of people.
Now you can say, “Alexa, I like this song,” and a device playing music in your home will lower the volume and reply, “OK. Rating sa
Natural language processing helps computers communicate with humans in their own language and scales other language-related tasks.
For example, NLP makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which par
Today’s machines can analyze more language-based data than humans, without fatigue and in a consistent, unbiased way.
Considering the staggering amount of unstructured data that’s generated every day, from medical records to social media, automatior
```

```
#word segmentation
words = nltk.word_tokenize(newdata)
print(words)
```

```
['While', 'natural', 'language', 'processing', 'isn', '’', 't', 'a', 'new', 'science', ',', 'the', 'technology', 'is', 'rapidly',
```

```
#Sentence segmentation
sentences = nltk.sent_tokenize(newdata)
for sentence in sentences:
    print(sentence)
    print()
```

```
While natural language processing isn’t a new science, the technology is rapidly advancing thanks to an increased interest in humar

As a human, you may speak and write in English, Spanish or Chinese.

But a computer’s native language – known as machine code or machine language – is largely incomprehensible to most people.

At your device’s lowest levels, communication occurs not with words but through millions of zeros and ones that produce logical act

Indeed, programmers used punch cards to communicate with the first computers 70 years ago.

This manual and arduous process was understood by a relatively small number of people.

Now you can say, “Alexa, I like this song,” and a device playing music in your home will lower the volume and reply, “OK.

Rating saved,” in a humanlike voice.

Then it adapts its algorithm to play that song – and others like it – the next time you listen to that music station.

Natural language processing helps computers communicate with humans in their own language and scales other language-related tasks.

For example, NLP makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which par

Today’s machines can analyze more language-based data than humans, without fatigue and in a consistent, unbiased way.

Considering the staggering amount of unstructured data that’s generated every day, from medical records to social media, automatior
```

```
# Convert to Lowercase
print(newdata.lower())
```

```
while natural language processing isn’t a new science, the technology is rapidly advancing thanks to an increased interest in humar
as a human, you may speak and write in english, spanish or chinese.
but a computer’s native language – known as machine code or machine language – is largely incomprehensible to most people.
at your device’s lowest levels, communication occurs not with words but through millions of zeros and ones that produce logical act
indeed, programmers used punch cards to communicate with the first computers 70 years ago.
this manual and arduous process was understood by a relatively small number of people.
now you can say, “alexa, i like this song,” and a device playing music in your home will lower the volume and reply, “ok. rating sa
natural language processing helps computers communicate with humans in their own language and scales other language-related tasks.
for example, nlp makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which par
today’s machines can analyze more language-based data than humans, without fatigue and in a consistent, unbiased way.
considering the staggering amount of unstructured data that’s generated every day, from medical records to social media, automatior
```

```
#Stop words removal

stop_words = set(stopwords.words('english'))
word_tokens = word_tokenize(newdata)
# converts the words in word_tokens to lower case and then checks whether
#they are present in stop_words or not
filtered_sentence = [w for w in word_tokens if not w.lower() in stop_words]
#with no lower case conversion
filtered_sentence = []

for w in word_tokens:
```

```
    if w not in stop_words:
        filtered_sentence.append(w)

print(word_tokens)
print("\nSentence after removing stop words:\n")
print(filtered_sentence)
```

```
['While', 'natural', 'language', 'processing', 'isn', '’', 't', 'a', 'new', 'science', ',', 'the', 'technology', 'is', 'rapidly',

Sentence after removing stop words:

['While', 'natural', 'language', 'processing', '’', 'new', 'science', ',', 'technology', 'rapidly', 'advancing', 'thanks', 'increas
```

```
#Stemming

ps = PorterStemmer()
words = ["process", "processing", "processed", "processers", "processes"]

for w in words:
    print(w, " : ", ps.stem(w))
```

```
process  :  process
processing  :  process
processed  :  process
processers  :  process
processes  :  process
```

```
# Lemmatization

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

print("communications :", lemmatizer.lemmatize("communications"))
print("programmers :", lemmatizer.lemmatize("programmers"))

# a denotes adjective in "pos"
print("lowest :", lemmatizer.lemmatize("lowest", pos ="a"))
```

```
communications : communication
programmers : programmer
lowest : low
```

```
# Part of speech tagger

stop_words = set(stopwords.words('english'))
tokenized = sent_tokenize(newdata)
for i in tokenized:

    # Word tokenizers is used to find the words
    # and punctuation in a string
    wordsList = nltk.word_tokenize(i)

    # removing stop words from wordList
    wordsList = [w for w in wordsList if not w in stop_words]

    #  Using a Tagger. Which is part-of-speech
    # tagger or POS-tagger.
    tagged = nltk.pos_tag(wordsList)

    print(tagged)
```

```
[('While', 'IN'), ('natural', 'JJ'), ('language', 'NN'), ('processing', 'NN'), ('’', 'JJ'), ('new', 'JJ'), ('science', 'NN'), (',',
[('As', 'IN'), ('human', 'JJ'), (',', ','), ('may', 'MD'), ('speak', 'VB'), ('write', 'JJ'), ('English', 'NNP'), (',', ','), ('Spar
[('But', 'CC'), ('computer', 'NN'), ('’', 'VBP'), ('native', 'JJ'), ('language', 'NN'), ('–', 'VBD'), ('known', 'VBN'), ('machine',
[('At', 'IN'), ('device', 'NN'), ('’', 'CD'), ('lowest', 'NN'), ('levels', 'NNS'), (',', ','), ('communication', 'NN'), ('occurs',
[('Indeed', 'RB'), (',', ','), ('programmers', 'NNS'), ('used', 'VBD'), ('punch', 'JJ'), ('cards', 'NNS'), ('communicate', 'VBP'),
[('This', 'DT'), ('manual', 'JJ'), ('arduous', 'JJ'), ('process', 'NN'), ('understood', 'VBD'), ('relatively', 'RB'), ('small', 'JJ
[('Now', 'RB'), ('say', 'VBP'), (',', ','), ('“', 'JJ'), ('Alexa', 'NNP'), (',', ','), ('I', 'PRP'), ('like', 'VBP'), ('song', 'RB
[('Rating', 'VBG'), ('saved', 'VBN'), (',', ','), ('”', 'FW'), ('humanlike', 'JJ'), ('voice', 'NN'), ('.', '.')]
[('Then', 'RB'), ('adapts', 'VBZ'), ('algorithm', 'JJ'), ('play', 'NN'), ('song', 'NN'), ('–', 'JJ'), ('others', 'NNS'), ('like',
[('Natural', 'JJ'), ('language', 'NN'), ('processing', 'NN'), ('helps', 'VBZ'), ('computers', 'NNS'), ('communicate', 'VBP'), ('hum
[('For', 'IN'), ('example', 'NN'), (',', ','), ('NLP', 'NNP'), ('makes', 'VBZ'), ('possible', 'JJ'), ('computers', 'NNS'), ('read',
[('Today', 'NN'), ('’', 'NNP'), ('machines', 'NNS'), ('analyze', 'VBP'), ('language-based', 'JJ'), ('data', 'NNS'), ('humans', 'NNS
[('Considering', 'VBG'), ('staggering', 'VBG'), ('amount', 'NN'), ('unstructured', 'JJ'), ('data', 'NNS'), ('’', 'RB'), ('generated
```