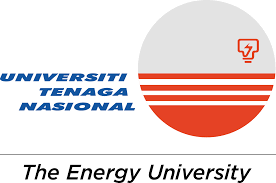
****

**CISB5123 Text Analytics**

**Lab 4**

**Basic Text Pre-processing**

Text pre-processing is a process to clean and prepare the textual data before they can be used as the input for text modelling techniques.

It consists of breaking down the **documents** in **corpus** into **tokens**.

**The steps in text pre-processing in Python**

1. Read the source data
2. Remove punctuations and standardize words into lowercases in the documents using **string** library
3. Remove numbers using **re.sub ( )** in **regular expression** library
4. Break down the words in documents into tokens using **nltk** library
5. Remove stopwords using **nltk** library
6. Perform word stemming using **Porter Stemmer** in **nltk** library
7. Perform word lemmatization using **WordNetLemmatizer( )** in **nltk** library

**Step 1: Read the source data**

*#import pandas module for creating dataframe*

**import** pandas **as** pd

*#read CSV into DataFrame*

data = pd.read\_csv("Review.csv")

data

A text on a screen

Description automatically generated

**Step 2: Remove punctuations and standardize words into lowercases in the documents using string library**

*#import string module for string manipulation*

**import** string

*#The constant in string module to remove punctuations*

string.punctuation



*#defining the function to remove punctuations in the documents*

**def** remove\_punctuation(text):

*# Initialize an empty string to store the result*

punctuation\_free = ""

*# Iterate over each character in the text*

**for** i **in** text:

*# Check if the character is not in the string.punctuation set*

**if** i **not in** string.punctuation:

*# If not, add the character to the result string*

punctuation\_free += i

**return** punctuation\_free

*#applying the remove\_punctuation function to the 'Review' column and storing the result in a new column 'clean\_punctuation'*

data['clean\_punctuation']= data['Review'].apply(remove\_punctuation)

data

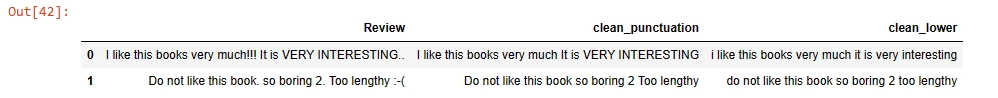
**A screenshot of a computer

Description automatically generated**

*#to standardize the cases in the documents into lower case*

data['clean\_lower']= data['clean\_punctuation'].str.lower()

data



**Step 3: Remove numbers using re.sub ( ) in regular expression library**

*#import regular expression library*

**import** re

*#function to remove digit (\d) or hypens (-) from the documents with an empty string ''*

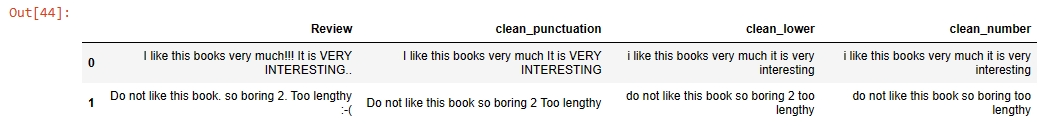
**def** remove\_numbers(text):

**return** re.sub("[\d-]",'',text)

*#applying the remove\_numbers function to the 'clean\_lower' column and storing the result in a new column 'clean\_number'*

data['clean\_number'] = data['clean\_lower'].apply(remove\_numbers)

data



**Step 4: Break down the words in documents into tokens using nltk library**

*#import Natural Language Processing (NLP) library called*

*#Natural Language Toolkit (NLTK)*

**import** nltk

nltk.download('punkt')

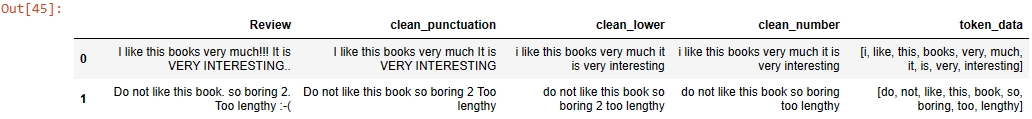
*# import the library for word tokenization*

**from** nltk.tokenize **import** word\_tokenize

*#the word tokens in the document*

data['token\_data']**=** data['clean\_number'].apply(word\_tokenize)

data



**Step 5: Remove stopwords using nltk library**

*#download stopwords*

nltk.download('stopwords')

*#Get the list of English stop words present in the library*

stopwords **=** nltk.corpus.stopwords.words('english')

*# Print the list of stopwords*

print(stopwords)

A close-up of a text

Description automatically generated

*#defining the function to remove stopwords from tokenized text*

**def** remove\_stopwords(text):

output = []

**for** i **in** text:

**if** i **not in** stopwords:

output.append(i)

**return** output

*#Applying the remove\_stopwords function to the 'token\_data' column and storing the result in a new column 'clean\_xstopwords'*

data['clean\_xstopwords'] = data['token\_data'].apply(remove\_stopwords)

data

A close up of a white background

Description automatically generated

**Step 6: Perform word stemming using Porter Stemmer in nltk library**

*#importing the Stemming function from nltk library*

**from** nltk.stem.porter **import** PorterStemmer

*#defining the object for stemming*

porter\_stemmer **=** PorterStemmer()

*#defining a function for stemming*

**def** stemming(text):

stem\_text = []

**for** word **in** text:

stemmed\_word = porter\_stemmer.stem(word)

stem\_text.append(stemmed\_word)

**return** stem\_text

*#applying the stemming function to the 'clean\_xstopwords' column and storing the result in a new column 'clean\_stemmed'*

data['clean\_stemmed'] **=** data['clean\_xstopwords'].apply(stemming)

data

A white background with black text

Description automatically generated

**Step 7: Perform word lemmatization using WordNetLemmatizer( ) in nltk library**

nltk.download('wordnet')

*#importing the Lemmatizer function from nltk library*

**from** nltk.stem **import** WordNetLemmatizer

*#defining the object for Lemmatization*

wordnet\_lemmatizer **=** WordNetLemmatizer()

*#defining the function for lemmatization*

**def** lemmatizer(text):

lemm\_text = []

**for** word **in** text:

lemmatized\_word = wordnet\_lemmatizer.lemmatize(word)

lemm\_text.append(lemmatized\_word)

**return** lemm\_text

*#* *#applying the lemmatizer function to the 'clean\_xstopwords' column and storing the result in a new column 'clean\_lemmatized1'*

data['clean\_lemmatized1']**=**data['clean\_xstopwords'].apply(lemmatizer)

*#* *#applying the lemmatizer function to the 'clean\_stemmed' column and storing the result in a new column 'clean\_lemmatized2'*

data['clean\_lemmatized2']**=**data['clean\_stemmed'].apply(lemmatizer)

data

A close-up of a chart

Description automatically generated