**Lab 2**

1. **Tokenization**

* The process of converting sentences or documents into a set of tokens or words.
* words are often separated by whitespaces but can also be separated by special characters such as a comma, a hyphen, etc

**1. a Tokenization by Splitting the Sentence by Whitespaces**

* Python provides a built-in function split() to separate sentences by character.
* split() separates the sentence by whitespace.
* Take any sentence of your choice and perform splitting sentence by whitespace.
* The split() function does not remove special characters, such as commas,

**1. b Tokenization using TextBlob Library**

* [TextBlob](https://textblob.readthedocs.io/en/dev/) is a Python library widely used in processing text data for NLP.
* . Before using the TextBlob library, make sure that the textblob and nltk
* Define a tokenize function using python to read a content from file.
* For every line in file Function should use word function in TextBlob to return list of tokens

### 1. c **Tokenization using NLTK Library**

* [NLTK](https://www.nltk.org/) library is yet another popular library for NLP applications
* Sentence tokenization: use sent\_tokenize function
* Word tokenization: use word\_tokenize function
* Define a tokenize function using python to read a content from file.
* For every line in file ,Function should use word function in sent\_tokenize and word\_tokenize to return list of tokens

## **2. Stemming**

* Stemming is a process of converting such words to their root form.
* Stemming does so by deleting the suffix and (or) prefix based on predefined rules
* NLTK library provides us with porter algorithms for stemming
* From nltk stem libaray get PorterStemmer
* Define a stemmer function using python to read a content from file.
* For every word in file ,Function should form word list.
* And it has to perform stemming by calling strip function in stem
* Print the word list and corresponding stem list.

## **3. Lemmatization**

* stemming sometimes results in meaningless words
* it does not take care if the word is a noun, verb, or adjective
* Lemmatization overcomes these drawbacks as it converts the words into a meaningful base word called the lemma.
* It takes vocabulary and morphological analysis into consideration rather than just trimming the words.
* NLTK library provides the WordNetLemmatizer to perform the lemmatization. The lemmatize() function accepts a word as an argument and returns its lemmatized form.
* Define a function lemmatize
* Function should read a file and it has return list of words after performing lemmatization using lemmatize function.

4. POS Tagging

* words can mean differently in different forms
* Word can mean differently when it is used as a noun and when it is used as a verb.
* Part of Speech tagging enables us to identify the form of a word in a sentence.
* NLTK has a built-in function pos\_tag() that accepts a list of words and returns the POS tag of the word
* you would need the wordnet from nltk.corpus to generate the POS tag correctly.
* The part of the speech tag is helpful in correctly lemmatizing the word

word\_lst = []

def lemmatizer(file):

lem\_lst = []

lem = WordNetLemmatizer()

f = open(file, 'r')

for l in f:

word\_lst.append(l.strip())

w = lem.lemmatize(str(l.strip()))

lem\_lst.append(w)

return lem\_lst

def generate\_tag(w):

t = nltk.pos\_tag([w])[0][1][0].upper()

dic = {

'N': wordnet.NOUN,

'V': wordnet.VERB,

'A': wordnet.ADJ,

'R': wordnet.ADV

}

return dic.get(t, wordnet.VERB)

lem\_lst = lemmatizer('reviews.txt')

for i in range(len(word\_lst)):

print(word\_lst[i]+"-->"+lem\_lst[i])