VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

NLP

FALL 2020-21 : DA

Name : Shounak Bhattacharya **Reg. No.:** 18BCE0548

My role was to work on the pre-processing part of NLP.

Pre-processing was performed right at the beginning of the task so that people could utilize it for further functionalities.

I initially worked on uploading the text files converting them into list format and stored them.

```
#initially i have use 4 files. Later to show an instantiation another t
ext file was created with a copy of text six years and counting new whe
re the file was roled on spinbot
#this was to show a cosine similarity instance where words with similar
 meaning but different ways of expressing were put
text1 = open('GettingSaucyAboutFood.txt','r').read()
text2 = open('SixYearsAndCounting.txt','r').read()
text3 = open('TrainToNowhere.txt','r').read()
text4 = open('WhatDreamsMayCome.txt','r').read()

list_texts = [text1, text2, text3, text4]
```

Our job was to make sure that all the data was tokenised.

```
#tokenising the text files
from nltk import word_tokenize
words0 = word_tokenize(doc_0)
words1 = word_tokenize(doc_1)
words2 = word_tokenize(doc_2)
words3 = word_tokenize(doc_3)
words4 = word_tokenize(doc_4)

#Conversion to lower case for cosine similarities and future convenienc
e
print(words0,"\n",words1,"\n",words2,"\n",words3,"\n",words4,"\n")
words0_new = [word.lower() for word in words0]
words1_new = [word.lower() for word in words1]
words2_new = [word.lower() for word in words2]
words3_new = [word.lower() for word in words3]
words4_new = [word.lower() for word in words4]
print(words0_new,"\n",words1_new,"\n",words2_new,"\n",words3_new,"\n",w
ords4_new)
```

```
#tokeninzing with word boundaries may cause an issue so we can remove t
he punctuations
```

I then converted all the tokenised words into lower case.

```
import string
print(string.punctuation)
```

Then I highlighted the necessity of removing punctuations.

```
text_p_1 = " ".join([char for char in words1_new if char not in string.
punctuation])
text_p_2 = " ".join([char for char in words2_new if char not in string.
punctuation])
text_p_3 = " ".join([char for char in words3_new if char not in string.
punctuation])
text_p_4 = " ".join([char for char in words4_new if char not in string.
punctuation])
```

The next task was post removing punctuations the data was again converted into a data stash. Hence tokenizing was a necessity.

```
words1_t = word_tokenize(text_p_1)
words2_t = word_tokenize(text_p_2)
words3_t = word_tokenize(text_p_3)
words4_t = word_tokenize(text_p_4)
print(words1_t,"\n",words2_t,"\n",words3_t,"\n",words4_t)
```

The next necessity was to remove stopwords like I, me, myself, we our etc. Stopwords are commonplace words which also offer lesser significance in the scheme of context.

```
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(stop_words)
words1f = [word for word in words1_t if word not in stop_words]
words2f = [word for word in words2_t if word not in stop_words]
words3f = [word for word in words3_t if word not in stop_words]
words4f = [word for word in words4_t if word not in stop_words]
print(words1f,"\n",words2f,"\n",words3f,"\n",words4f)
```

We used Lancaster Stemmer over a Porter Stemmer for the fundamental reason that Lancaster is much more penetrative than Porter. It significantly can curtail down your dataset.

```
from nltk import LancasterStemmer
lc = LancasterStemmer()
```

```
stemmed1 = [lc.stem(word) for word in words1f]
stemmed2 = [lc.stem(word) for word in words2f]
stemmed3 = [lc.stem(word) for word in words3f]
stemmed4 = [lc.stem(word) for word in words4f]

print(stemmed1,"\n", stemmed2,"\n", stemmed3,"\n", stemmed4)
```

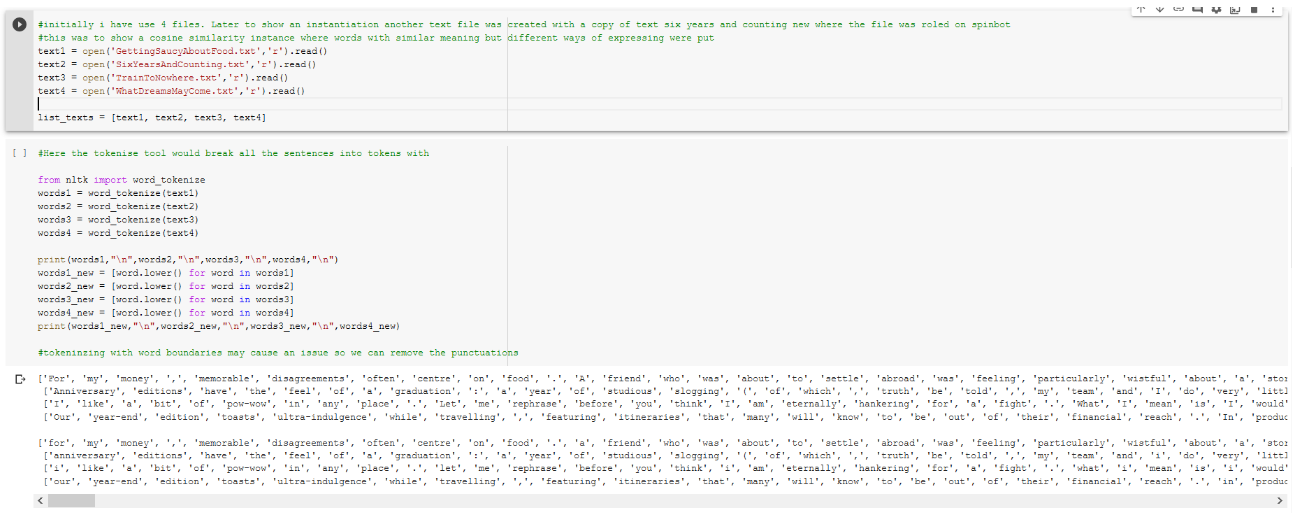
Finally, I added Part of speech tagging. Hence we can now make sure that the data has been pre processed for further work.

```
from nltk import pos_tag
pos1 = pos_tag(stemmed1)
pos2 = pos_tag(stemmed2)
pos3 = pos_tag(stemmed3)
pos4 = pos_tag(stemmed4)
print(pos1,"\n", pos2,"\n", pos3,"\n", pos4)
```

A series of images have been uploaded below besides the colab file that has been shared.

(This file is only a preprocessing module and the entire things has been moderated and changed in the mainfile to normalize all requirements.)

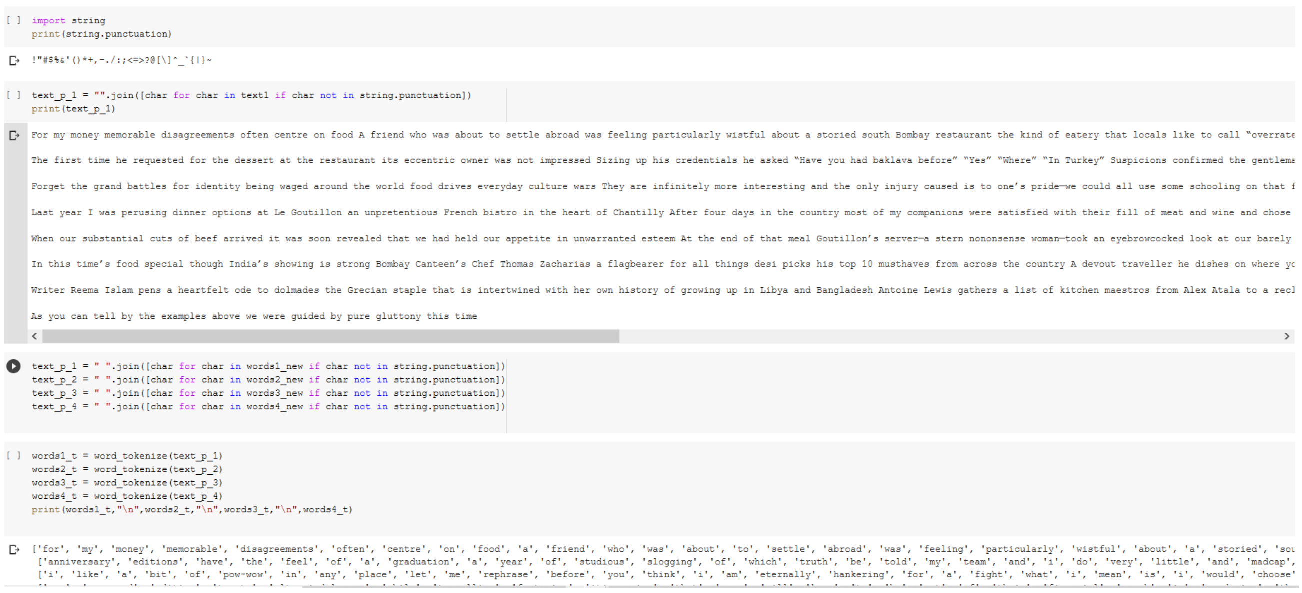
An additional file was added to the 4 given after turning it on spinbot to compare the efficiency of the cosine similarities and the jaccard similarity.



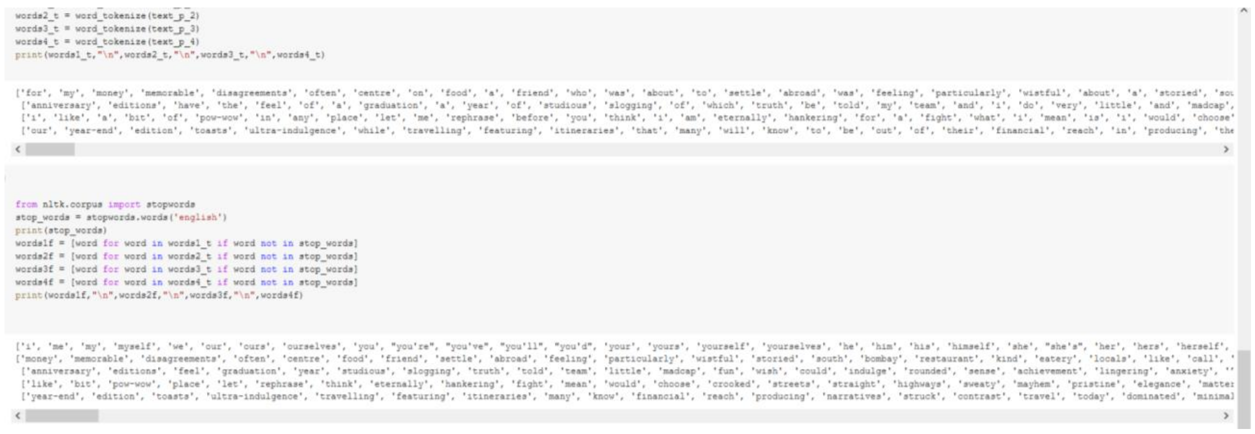
```
#initially i have use 4 files. Later to show an instantiation another text file was created with a copy of text six years and counting new where the file was roled on spinbot
#this was to show a cosine similarity instance where words with similar meaning but different ways of expressing were put
text1 = open('GettingSaucyAboutFood.txt','r').read()
text2 = open('SixYearsAndCounting.txt','r').read()
text3 = open('TrainToNowhere.txt','r').read()
text4 = open('WhatDreamsMayCome.txt','r').read()

list_texts = [text1, text2, text3, text4]
```

```
[ ] #Here the tokenise tool would break all the sentences into tokens with

from nltk import word_tokenize
words1 = word_tokenize(text1)
words2 = word_tokenize(text2)
words3 = word_tokenize(text3)
words4 = word_tokenize(text4)

print(words1,"\n",words2,"\n",words3,"\n",words4,"\n")
words1_new = [word.lower() for word in words1]
words2_new = [word.lower() for word in words2]
words3_new = [word.lower() for word in words3]
words4_new = [word.lower() for word in words4]
print(words1_new,"\n",words2_new,"\n",words3_new,"\n",words4_new)

#tokeninzing with word boundaries may cause an issue so we can remove the punctuations
```

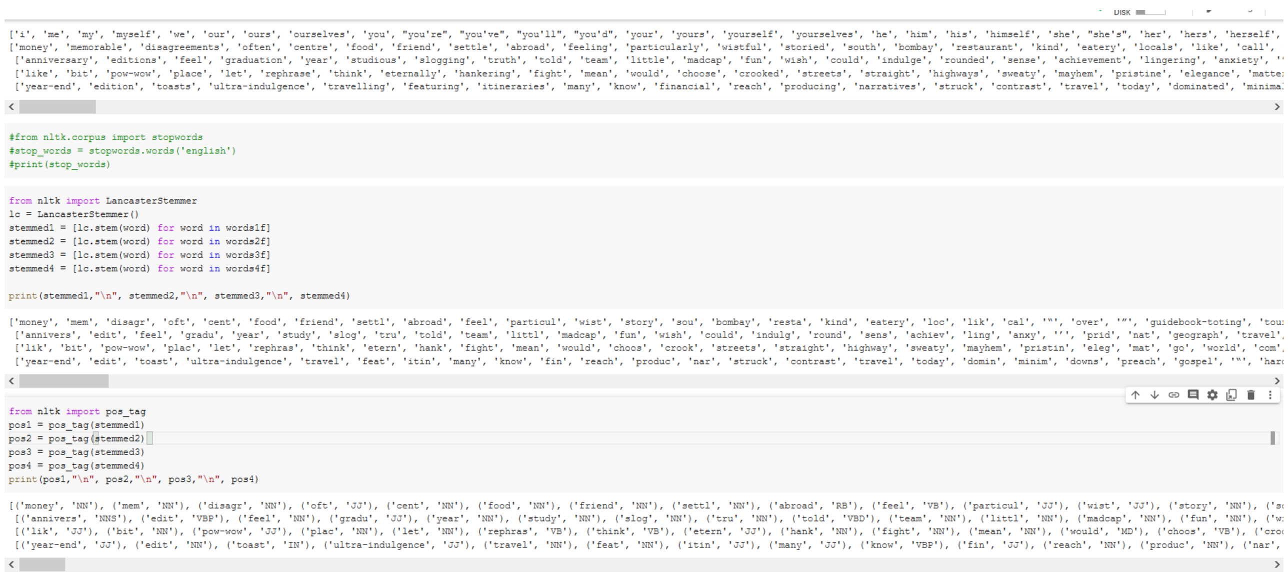
```
['For', 'my', 'money', ',', 'memorable', 'disagreements', 'often', 'centre', 'on', 'food', '.', 'A', 'friend', 'who', 'was', 'about', 'to', 'settle', 'abroad', 'was', 'feeling', 'particularly', 'wistful', 'about', 'a', 'stor
 ['Anniversary', 'editions', 'have', 'the', 'feel', 'of', 'a', 'graduation', ':', 'a', 'year', 'of', 'studious', 'slogging', '(', 'of', 'which', ',', 'truth', 'be', 'told', ',', 'my', 'team', 'and', 'I', 'do', 'very', 'littl
 ['I', 'like', 'a', 'bit', 'of', 'pow-wow', 'in', 'any', 'place', '.', 'Let', 'me', 'rephrase', 'before', 'you', 'think', 'I', 'am', 'eternally', 'hankering', 'for', 'a', 'fight', '.', 'What', 'I', 'mean', 'is', 'I', 'would'
 ['Our', 'year-end', 'edition', 'toasts', 'ultra-indulgence', 'while', 'travelling', ',', 'featuring', 'itineraries', 'that', 'many', 'will', 'know', 'to', 'be', 'out', 'of', 'their', 'financial', 'reach', '.', 'In', 'produc

['for', 'my', 'money', ',', 'memorable', 'disagreements', 'often', 'centre', 'on', 'food', '.', 'a', 'friend', 'who', 'was', 'about', 'to', 'settle', 'abroad', 'was', 'feeling', 'particularly', 'wistful', 'about', 'a', 'stor
 ['anniversary', 'editions', 'have', 'the', 'feel', 'of', 'a', 'graduation', ':', 'a', 'year', 'of', 'studious', 'slogging', '(', 'of', 'which', ',', 'truth', 'be', 'told', ',', 'my', 'team', 'and', 'i', 'do', 'very', 'littl
 ['i', 'like', 'a', 'bit', 'of', 'pow-wow', 'in', 'any', 'place', '.', 'let', 'me', 'rephrase', 'before', 'you', 'think', 'i', 'am', 'eternally', 'hankering', 'for', 'a', 'fight', '.', 'what', 'i', 'mean', 'is', 'i', 'would'
 ['our', 'year-end', 'edition', 'toasts', 'ultra-indulgence', 'while', 'travelling', ',', 'featuring', 'itineraries', 'that', 'many', 'will', 'know', 'to', 'be', 'out', 'of', 'their', 'financial', 'reach', '.', 'in', 'produc
```



```
[ ] import string
    print(string.punctuation)

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

[ ] text_p_1 = "".join([char for char in text1 if char not in string.punctuation])
    print(text_p_1)

For my money memorable disagreements often centre on food A friend who was about to settle abroad was feeling particularly wistful about a storied south Bombay restaurant the kind of eatery that locals like to call “overrate

The first time he requested for the dessert at the restaurant its eccentric owner was not impressed Sizing up his credentials he asked “Have you had baklava before” “Yes” “Where” “In Turkey” Suspicions confirmed the gentlema

Forget the grand battles for identity being waged around the world food drives everyday culture wars They are infinitely more interesting and the only injury caused is to one’s pride—we could all use some schooling on that f

Last year I was perusing dinner options at Le Goutillon an unpretentious French bistro in the heart of Chantilly After four days in the country most of my companions were satisfied with their fill of meat and wine and chose

When our substantial cuts of beef arrived it was soon revealed that we had held our appetite in unwarranted esteem At the end of that meal Goutillon’s server—a stern nononsense woman—took an eyebrowcocked look at our barely

In this time’s food special though India’s showing is strong Bombay Canteen’s Chef Thomas Zacharias a flagbearer for all things desi picks his top 10 musthaves from across the country A devout traveller he dishes on where yo

Writer Reema Islam pens a heartfelt ode to dolmades the Grecian staple that is intertwined with her own history of growing up in Libya and Bangladesh Antoine Lewis gathers a list of kitchen maestros from Alex Atala to a recl

As you can tell by the examples above we were guided by pure gluttony this time

text_p_1 = " ".join([char for char in words1_new if char not in string.punctuation])
text_p_2 = " ".join([char for char in words2_new if char not in string.punctuation])
text_p_3 = " ".join([char for char in words3_new if char not in string.punctuation])
text_p_4 = " ".join([char for char in words4_new if char not in string.punctuation])

[ ] words1_t = word_tokenize(text_p_1)
    words2_t = word_tokenize(text_p_2)
    words3_t = word_tokenize(text_p_3)
    words4_t = word_tokenize(text_p_4)
    print(words1_t,"\n",words2_t,"\n",words3_t,"\n",words4_t)

['for', 'my', 'money', 'memorable', 'disagreements', 'often', 'centre', 'on', 'food', 'a', 'friend', 'who', 'was', 'about', 'to', 'settle', 'abroad', 'was', 'feeling', 'particularly', 'wistful', 'about', 'a', 'storied', 'sou
 ['anniversary', 'editions', 'have', 'the', 'feel', 'of', 'a', 'graduation', 'a', 'year', 'of', 'studious', 'slogging', 'of', 'which', 'truth', 'be', 'told', 'my', 'team', 'and', 'i', 'do', 'very', 'little', 'and', 'madcap',
 ['i', 'like', 'a', 'bit', 'of', 'pow-wow', 'in', 'any', 'place', 'let', 'me', 'rephrase', 'before', 'you', 'think', 'i', 'am', 'eternally', 'hankering', 'for', 'a', 'fight', 'what', 'i', 'mean', 'is', 'i', 'would', 'choose'
```



```
words2_t = word_tokenize(text_p_2)
words3_t = word_tokenize(text_p_3)
words4_t = word_tokenize(text_p_4)
print(words1_t,"\n",words2_t,"\n",words3_t,"\n",words4_t)

['for', 'my', 'money', 'memorable', 'disagreements', 'often', 'centre', 'on', 'food', 'a', 'friend', 'who', 'was', 'about', 'to', 'settle', 'abroad', 'was', 'feeling', 'particularly', 'wistful', 'about', 'a', 'storied', 'sou
 ['anniversary', 'editions', 'have', 'the', 'feel', 'of', 'a', 'graduation', 'a', 'year', 'of', 'studious', 'slogging', 'of', 'which', 'truth', 'be', 'told', 'my', 'team', 'and', 'i', 'do', 'very', 'little', 'and', 'madcap',
 ['i', 'like', 'a', 'bit', 'of', 'pow-wow', 'in', 'any', 'place', 'let', 'me', 'rephrase', 'before', 'you', 'think', 'i', 'am', 'eternally', 'hankering', 'for', 'a', 'fight', 'what', 'i', 'mean', 'is', 'i', 'would', 'choose'
 ['our', 'year-end', 'edition', 'toasts', 'ultra-indulgence', 'while', 'travelling', 'featuring', 'itineraries', 'that', 'many', 'will', 'know', 'to', 'be', 'out', 'of', 'their', 'financial', 'reach', 'in', 'producing', 'the

from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(stop_words)
words1f = [word for word in words1_t if word not in stop_words]
words2f = [word for word in words2_t if word not in stop_words]
words3f = [word for word in words3_t if word not in stop_words]
words4f = [word for word in words4_t if word not in stop_words]
print(words1f,"\n",words2f,"\n",words3f,"\n",words4f)

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself',
['money', 'memorable', 'disagreements', 'often', 'centre', 'food', 'friend', 'settle', 'abroad', 'feeling', 'particularly', 'wistful', 'storied', 'south', 'bombay', 'restaurant', 'kind', 'eatery', 'locals', 'like', 'call', '
 ['anniversary', 'editions', 'feel', 'graduation', 'year', 'studious', 'slogging', 'truth', 'told', 'team', 'little', 'madcap', 'fun', 'wish', 'could', 'indulge', 'rounded', 'sense', 'achievement', 'lingering', 'anxiety', ''
 ['like', 'bit', 'pow-wow', 'place', 'let', 'rephrase', 'think', 'eternally', 'hankering', 'fight', 'mean', 'would', 'choose', 'crooked', 'streets', 'straight', 'highways', 'sweaty', 'mayhem', 'pristine', 'elegance', 'matter
 ['year-end', 'edition', 'toasts', 'ultra-indulgence', 'travelling', 'featuring', 'itineraries', 'many', 'know', 'financial', 'reach', 'producing', 'narratives', 'struck', 'contrast', 'travel', 'today', 'dominated', 'minimal
```



```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself',
['money', 'memorable', 'disagreements', 'often', 'centre', 'food', 'friend', 'settle', 'abroad', 'feeling', 'particularly', 'wistful', 'storied', 'south', 'bombay', 'restaurant', 'kind', 'eatery', 'locals', 'like', 'call', '
 ['anniversary', 'editions', 'feel', 'graduation', 'year', 'studious', 'slogging', 'truth', 'told', 'team', 'little', 'madcap', 'fun', 'wish', 'could', 'indulge', 'rounded', 'sense', 'achievement', 'lingering', 'anxiety', ''
 ['like', 'bit', 'pow-wow', 'place', 'let', 'rephrase', 'think', 'eternally', 'hankering', 'fight', 'mean', 'would', 'choose', 'crooked', 'streets', 'straight', 'highways', 'sweaty', 'mayhem', 'pristine', 'elegance', 'matte
 ['year-end', 'edition', 'toasts', 'ultra-indulgence', 'travelling', 'featuring', 'itineraries', 'many', 'know', 'financial', 'reach', 'producing', 'narratives', 'struck', 'contrast', 'travel', 'today', 'dominated', 'minima
```

```
#from nltk.corpus import stopwords
#stop_words = stopwords.words('english')
#print(stop_words)

from nltk import LancasterStemmer
lc = LancasterStemmer()
stemmed1 = [lc.stem(word) for word in words1f]
stemmed2 = [lc.stem(word) for word in words2f]
stemmed3 = [lc.stem(word) for word in words3f]
stemmed4 = [lc.stem(word) for word in words4f]

print(stemmed1,"\n", stemmed2,"\n", stemmed3,"\n", stemmed4)

['money', 'mem', 'disagr', 'oft', 'cent', 'food', 'friend', 'settl', 'abroad', 'feel', 'particul', 'wist', 'story', 'sou', 'bombay', 'resta', 'kind', 'eatery', 'loc', 'lik', 'cal', '“', 'over', '”', 'guidebook-toting', 'tou
 ['annivers', 'edit', 'feel', 'gradu', 'year', 'study', 'slog', 'tru', 'told', 'team', 'littl', 'madcap', 'fun', 'wish', 'could', 'indulg', 'round', 'sens', 'achiev', 'ling', 'anxy', '’', 'prid', 'nat', 'geograph', 'travel',
 ['lik', 'bit', 'pow-wow', 'plac', 'let', 'rephras', 'think', 'etern', 'hank', 'fight', 'mean', 'would', 'choos', 'crook', 'streets', 'straight', 'highway', 'sweaty', 'mayhem', 'pristin', 'eleg', 'mat', 'go', 'world', 'com',
 ['year-end', 'edit', 'toast', 'ultra-indulgence', 'travel', 'feat', 'itin', 'many', 'know', 'fin', 'reach', 'produc', 'nar', 'struck', 'contrast', 'travel', 'today', 'domin', 'minim', 'downs', 'preach', 'gospel', '“', 'har

from nltk import pos_tag
pos1 = pos_tag(stemmed1)
pos2 = pos_tag(stemmed2)
pos3 = pos_tag(stemmed3)
pos4 = pos_tag(stemmed4)
print(pos1,"\n", pos2,"\n", pos3,"\n", pos4)

[('money', 'NN'), ('mem', 'NN'), ('disagr', 'NN'), ('oft', 'JJ'), ('cent', 'NN'), ('food', 'NN'), ('friend', 'NN'), ('settl', 'NN'), ('abroad', 'RB'), ('feel', 'VB'), ('particul', 'JJ'), ('wist', 'JJ'), ('story', 'NN'), ('s
 [('annivers', 'NNS'), ('edit', 'VBP'), ('feel', 'NN'), ('gradu', 'JJ'), ('year', 'NN'), ('study', 'NN'), ('slog', 'NN'), ('tru', 'NN'), ('told', 'VBD'), ('team', 'NN'), ('littl', 'NN'), ('madcap', 'NN'), ('fun', 'NN'), ('w
 [('lik', 'JJ'), ('bit', 'NN'), ('pow-wow', 'JJ'), ('plac', 'NN'), ('let', 'NN'), ('rephras', 'VB'), ('think', 'VB'), ('etern', 'JJ'), ('hank', 'NN'), ('fight', 'NN'), ('mean', 'NN'), ('would', 'MD'), ('choos', 'VB'), ('cro
 [('year-end', 'JJ'), ('edit', 'NN'), ('toast', 'IN'), ('ultra-indulgence', 'JJ'), ('travel', 'NN'), ('feat', 'NN'), ('itin', 'JJ'), ('many', 'JJ'), ('know', 'VBP'), ('fin', 'JJ'), ('reach', 'NN'), ('produc', 'NN'), ('nar',
```

Later on, under further consideration we implemented lemmatization with some discussion with Aniket directly in the main document.

As decided by our group leader Shivam I and Navdeep were told to perform this task and under efficient collaboration with him we completed the entire task to the fullest.

The link of our work and task has been shared below and I have also given you sharing access for the same.

https://colab.research.google.com/drive/1uNv9RTnm5_-aVLXYL9v5zsNksA25Ew3S?usp=sharing

Conclusion

I would like to thank our teacher Sharmila Banu ma'am, our group leader Shivam and all my group mates for this really great experience. It enabled me to work in a group, learn new concepts and contribute a preprocessing module which is really essential to any NLP task.

I got to work in a team with some really great minds. Look forward to many more such tasks.

Auf Wiedersehen!