Day 68

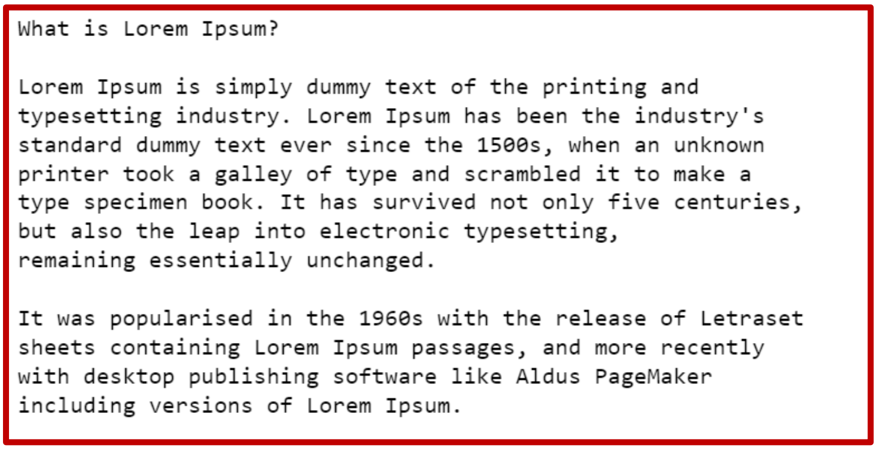
DIY

Q1. Problem Statement: NLP text processing

Write a Python program that reads the *demotext.txt* text file (provided on LMS). The following are the tasks that are to be taken into consideration while constructing the solution for text processing using the NLTK library.

1. Load the *demotext.txt* text file into a variable and then close the file
2. Do word-wise tokenization list out generated tokens
3. Transform each token into a small case
4. Remove stop words from the generated token list
5. Remove extra symbols like commas, full stops, and question marks using a regular expression tokenizer and store them in another variable
6. Do bigram and trigram for generated tokens

Input Table Format:



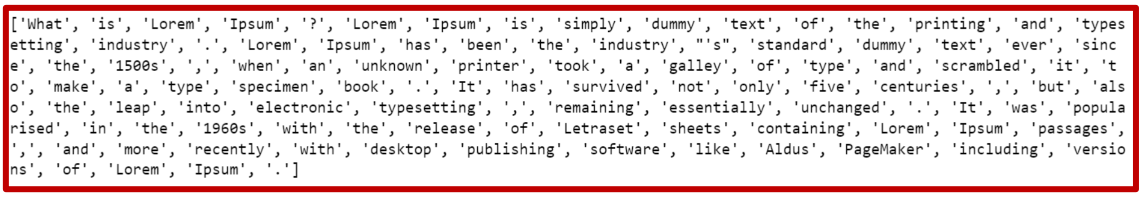
```
What is Lorem Ipsum?

Lorem Ipsum is simply dummy text of the printing and
typesetting industry. Lorem Ipsum has been the industry's
standard dummy text ever since the 1500s, when an unknown
printer took a galley of type and scrambled it to make a
type specimen book. It has survived not only five centuries,
but also the leap into electronic typesetting,
remaining essentially unchanged.

It was popularised in the 1960s with the release of Letraset
sheets containing Lorem Ipsum passages, and more recently
with desktop publishing software like Aldus PageMaker
including versions of Lorem Ipsum.
```

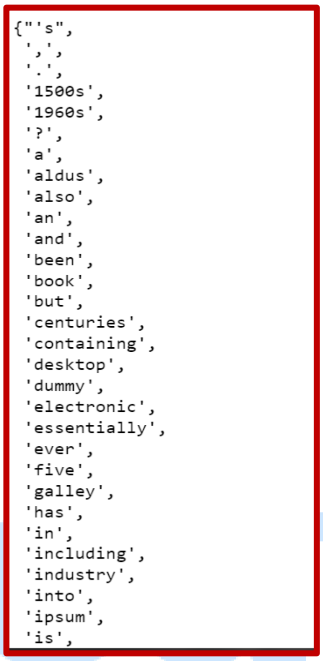
Sample Output:

2. Do word-wise tokenization list out generated tokens



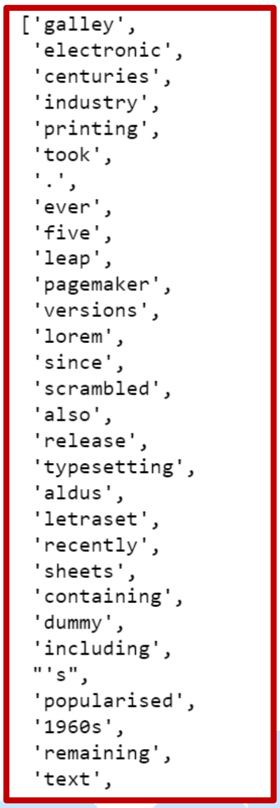
```
['What', 'is', 'Lorem', 'Ipsum', '?', 'Lorem', 'Ipsum', 'is', 'simply', 'dummy', 'text', 'of', 'the', 'printing', 'and', 'types
etting', 'industry', '.', 'Lorem', 'Ipsum', 'has', 'been', 'the', 'industry', "'s", 'standard', 'dummy', 'text', 'ever', 'sinc
e', 'the', '1500s', ',', 'when', 'an', 'unknown', 'printer', 'took', 'a', 'galley', 'of', 'type', 'and', 'scrambled', 'it', 't
o', 'make', 'a', 'type', 'specimen', 'book', '.', 'It', 'has', 'survived', 'not', 'only', 'five', 'centuries', ',', 'but', 'als
o', 'the', 'leap', 'into', 'electronic', 'typesetting', ',', 'remaining', 'essentially', 'unchanged', '.', 'It', 'was', 'popula
rised', 'in', 'the', '1960s', 'with', 'the', 'release', 'of', 'Letraset', 'sheets', 'containing', 'Lorem', 'Ipsum', 'passages',
',', 'and', 'more', 'recently', 'with', 'desktop', 'publishing', 'software', 'like', 'Aldus', 'PageMaker', 'including', 'versio
ns', 'of', 'Lorem', 'Ipsum', '.']
```

3. Transform each token into a small case



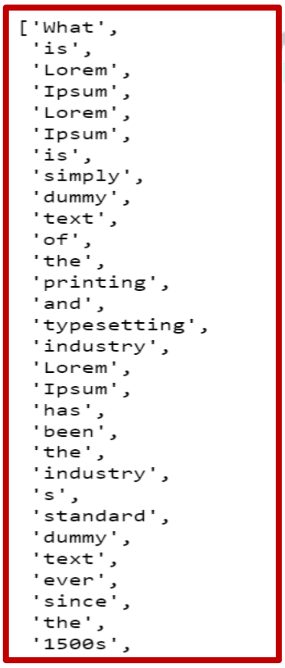
```
{"'s",
 ',',
 '.',
 '1500s',
 '1960s',
 '?',
 'a',
 'aldus',
 'also',
 'an',
 'and',
 'been',
 'book',
 'but',
 'centuries',
 'containing',
 'desktop',
 'dummy',
 'electronic',
 'essentially',
 'ever',
 'five',
 'galley',
 'has',
 'in',
 'including',
 'industry',
 'into',
 'ipsum',
 'is',
```

4. Remove stop words from the generated token list



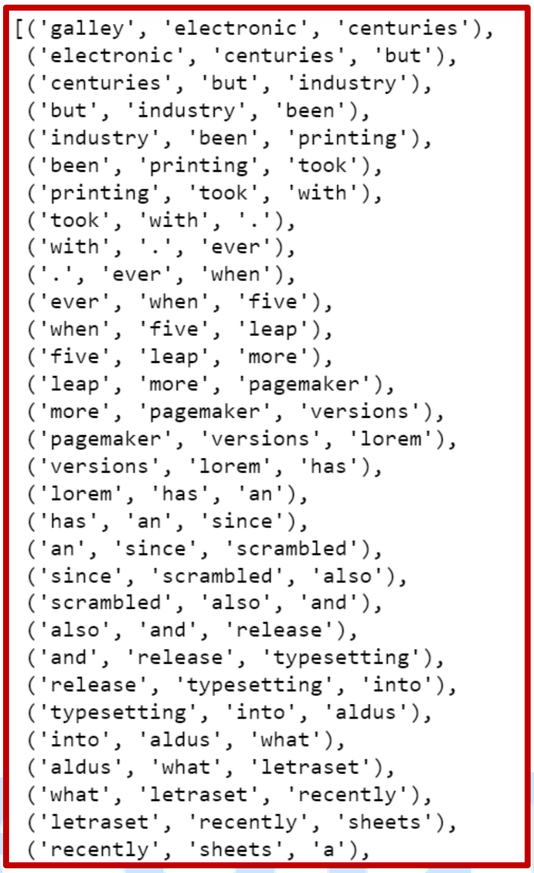
```
['galley',
 'electronic',
 'centuries',
 'industry',
 'printing',
 'took',
 '.',
 'ever',
 'five',
 'leap',
 'pagemaker',
 'versions',
 'lorem',
 'since',
 'scrambled',
 'also',
 'release',
 'typesetting',
 'aldus',
 'letraset',
 'recently',
 'sheets',
 'containing',
 'dummy',
 'including',
 "'s",
 'popularised',
 '1960s',
 'remaining',
 'text',
```

5. Remove extra symbols like commas, full stops, and question marks using a regular expression tokenizer and store them in another variable

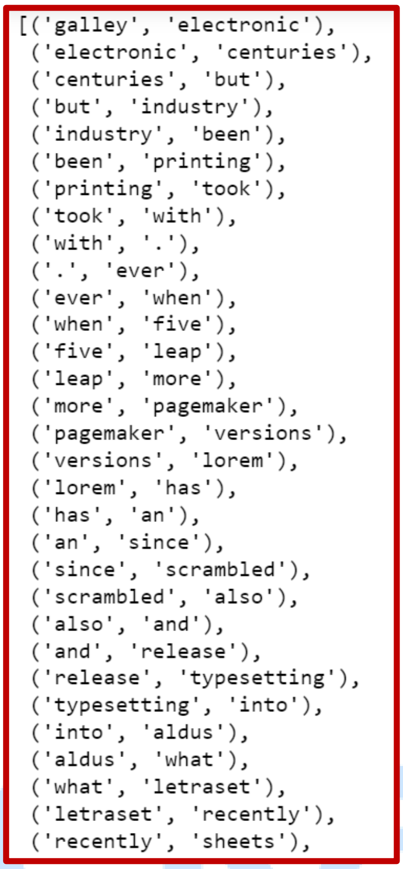


```
['What',
 'is',
 'Lorem',
 'Ipsum',
 'Lorem',
 'Ipsum',
 'is',
 'simply',
 'dummy',
 'text',
 'of',
 'the',
 'printing',
 'and',
 'typesetting',
 'industry',
 'Lorem',
 'Ipsum',
 'has',
 'been',
 'the',
 'industry',
 's',
 'standard',
 'dummy',
 'text',
 'ever',
 'since',
 'the',
 '1500s',
```

6. Do bigram and trigram for generated tokens



```
[('galley', 'electronic', 'centuries'),
 ('electronic', 'centuries', 'but'),
 ('centuries', 'but', 'industry'),
 ('but', 'industry', 'been'),
 ('industry', 'been', 'printing'),
 ('been', 'printing', 'took'),
 ('printing', 'took', 'with'),
 ('took', 'with', '.'),
 ('with', '.', 'ever'),
 ('.', 'ever', 'when'),
 ('ever', 'when', 'five'),
 ('when', 'five', 'leap'),
 ('five', 'leap', 'more'),
 ('leap', 'more', 'pagemaker'),
 ('more', 'pagemaker', 'versions'),
 ('pagemaker', 'versions', 'lorem'),
 ('versions', 'lorem', 'has'),
 ('lorem', 'has', 'an'),
 ('has', 'an', 'since'),
 ('an', 'since', 'scrambled'),
 ('since', 'scrambled', 'also'),
 ('scrambled', 'also', 'and'),
 ('also', 'and', 'release'),
 ('and', 'release', 'typesetting'),
 ('release', 'typesetting', 'into'),
 ('typesetting', 'into', 'aldus'),
 ('into', 'aldus', 'what'),
 ('aldus', 'what', 'letraset'),
 ('what', 'letraset', 'recently'),
 ('letraset', 'recently', 'sheets'),
 ('recently', 'sheets', 'a'),
```



```
[('galley', 'electronic'),
 ('electronic', 'centuries'),
 ('centuries', 'but'),
 ('but', 'industry'),
 ('industry', 'been'),
 ('been', 'printing'),
 ('printing', 'took'),
 ('took', 'with'),
 ('with', '.'),
 ('.', 'ever'),
 ('ever', 'when'),
 ('when', 'five'),
 ('five', 'leap'),
 ('leap', 'more'),
 ('more', 'pagemaker'),
 ('pagemaker', 'versions'),
 ('versions', 'lorem'),
 ('lorem', 'has'),
 ('has', 'an'),
 ('an', 'since'),
 ('since', 'scrambled'),
 ('scrambled', 'also'),
 ('also', 'and'),
 ('and', 'release'),
 ('release', 'typesetting'),
 ('typesetting', 'into'),
 ('into', 'aldus'),
 ('aldus', 'what'),
 ('what', 'letraset'),
 ('letraset', 'recently'),
 ('recently', 'sheets'),
```