NLP Assignment 2: Converting Words into Vectors (Basics techniques)_Practice Sheet

• Bagofwords

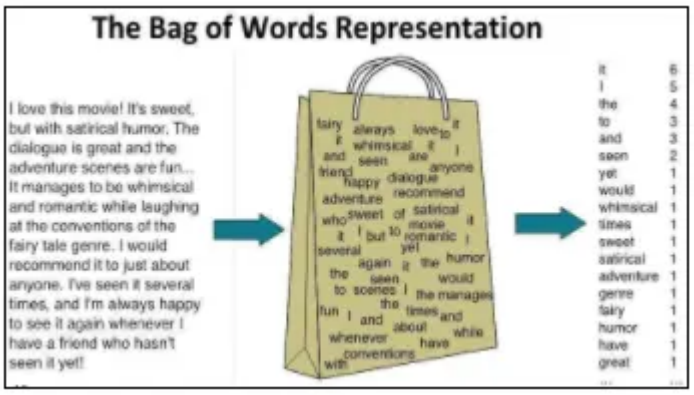
1. Bag of Words[BOG]
2. List item
3. TD-IDF
4. N-Grams
5. Part of speech[POS]

Bag of words

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval. In this model, a text (such as a sentence or a document) is represented as the bag of its words, disregarding grammar and even word order but keeping multiplicity.

The bag-of-words model is commonly used in methods of document classification where the occurrence of each word is used as a feature for training a classifier.

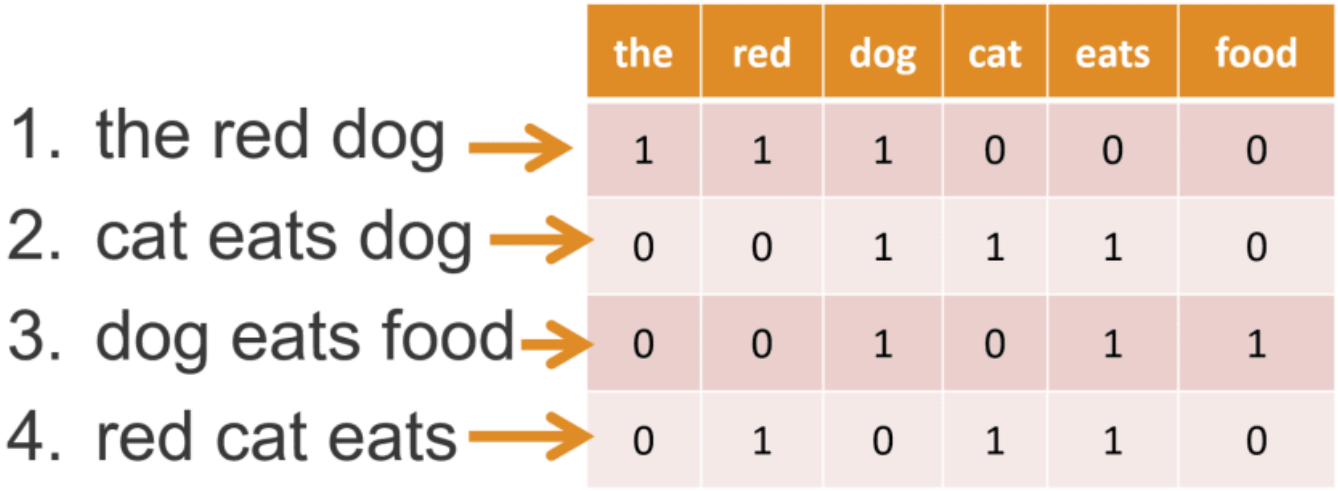
The Bag-of-words model is one example of a Vector space model.



CountVectorizer

CountVectorizer is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency of each word that occurs in the entire text.

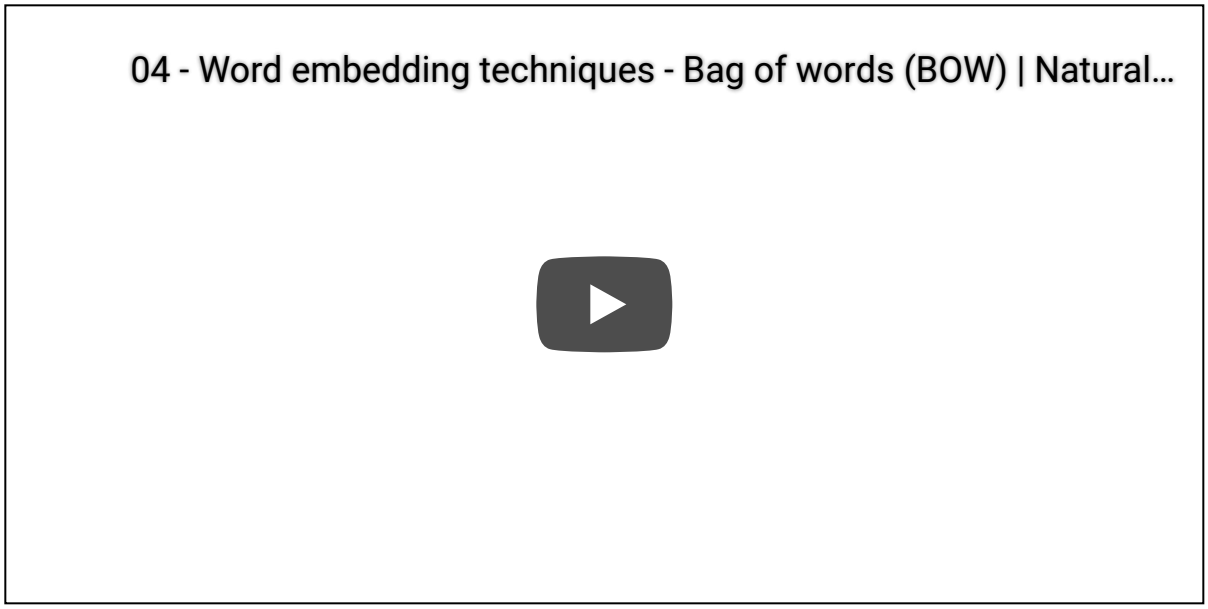
It creates a matrix in which each unique word is represented by a column of the matrix, and each text sample from the document is a row in the matrix.



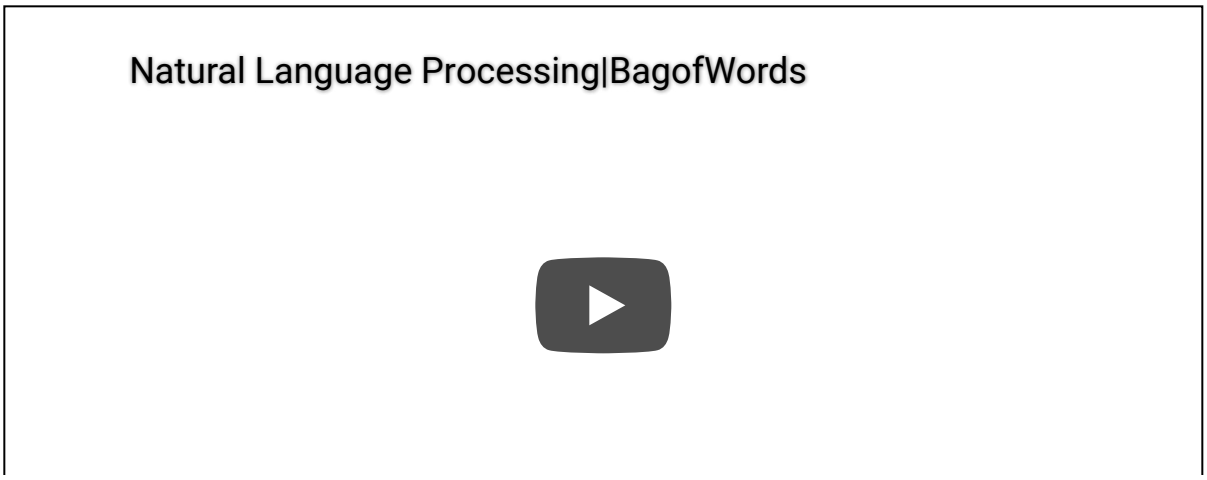
	the	red	dog	cat	eats	food
1. the red dog →	1	1	1	0	0	0
2. cat eats dog →	0	0	1	1	1	0
3. dog eats food →	0	0	1	0	1	1
4. red cat eats →	0	1	0	1	1	0

Documentation: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html

```
##### Refer Video
from IPython.display import YouTubeVideo
YouTubeVideo('L_v79r8Yqqs', width=600, height=300)
```




```
##### Refer Video
YouTubeVideo('iu2-G_5YkEo', width=600, height=300)
```




```
#importing libraries
# import nltk and download punkt,wordnet and stopwords
# your code here
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
# import re,stopwords,PorterStemmer and WordNetLemmatizer
# your code here
```

```
paragraph =  """Natural language processing (NLP) refers to the branch of computer science—an
NLP combines computational linguistics—rule-based modeling of human language—with statistical
NLP drives computer programs that translate text from one language to another, respond to spo
```

```
# create PorterStemmer and WordNetLemmatizer objects
# your code here
```

```
# tokenize the above paragraph into sentences
# your code here
```

```
'Natural language processing (NLP) refers to the branch of computer science—and more sp
ecifically, the branch of artificial intelligence or AI—concerned with giving computers
the ability to understand text and spoken words in much the same way human beings can.'
```

```
# fill in the missing places for the code block
listofsent= []
for i in range(len(___"your answer here"___)):
    wordfile = re.sub('[^a-zA-Z]', ' ', sentences[i])
    wordfile = # convert into lower case
```

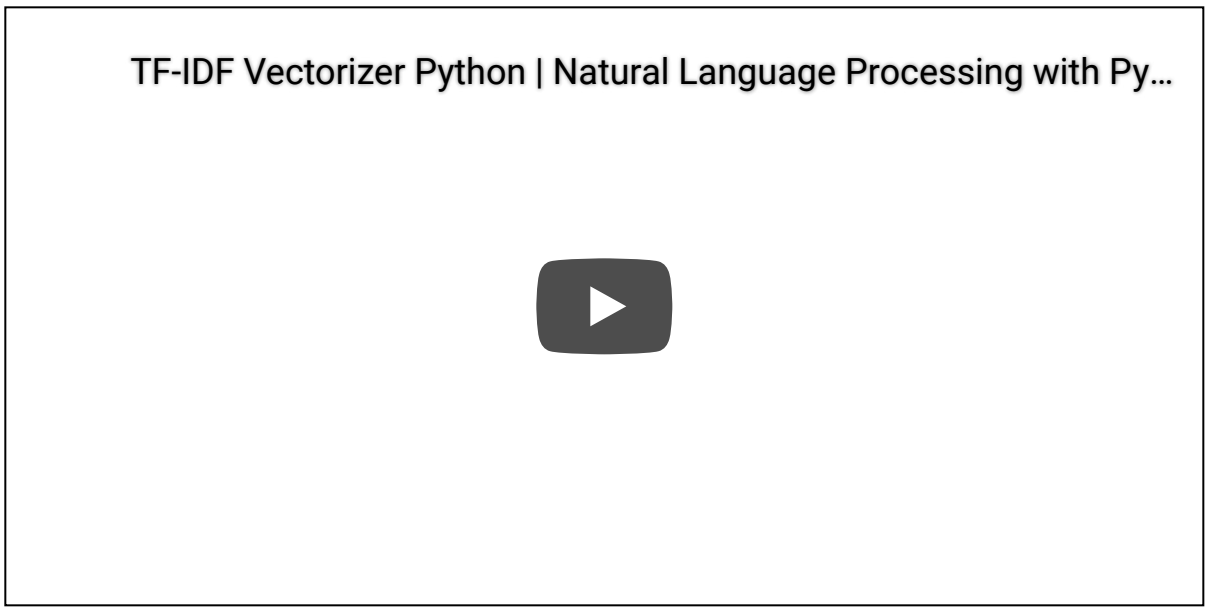
```
    wordfile = # split the wordfile
    wordfile = [ps."your answer here"(word) for word in wordfile if not word in set(stopwords
    wordfile = # join the wordfile with single blankspaces
    listofsent.append(wordfile)

# Create vectors of the above words using CountVectorizer
# your code here
```

TD-IDF

TF-IDF stands “Term Frequency — Inverse Document Frequency”.
This is a technique to quantify words in a set of documents. We generally compute a score for each word to signify its importance in the document and corpus. This method is a widely used technique in Information Retrieval and Text Mining.


```
##### Refer Video
YouTubeVideo('ZeQgNhc0vag', width=600, height=300)
```




```
##### Refer Video
YouTubeVideo('https://youtu.be/z9myrLOF_1M', width=600, height=300)
```

```
# importing libraries
# import nltk and download punkt,wordnet and stopwords
# your code here
```

```
    [nltk_data] Downloading package punkt to /root/nltk_data...
```

```
[nltk_data]   Package punkt 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 stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

```
# import re,stopwords,PorterStemmer and WordNetLemmatizer
# your code here
```

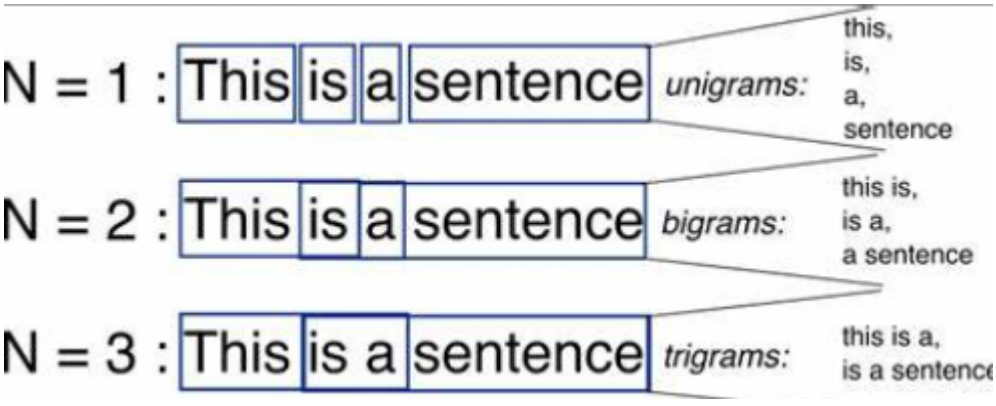
```
# create PorterStemmer and WordNetLemmatizer objects
# your code here
# tokenize the above paragraph into sentences
# your code here
```

```
# fill in the missing places for the code block
listofsent2= []
for i in range(len(___"your answer here"___)):
    wordfile1 = re.sub('[^a-zA-Z]', ' ', sentences1[i])
    wordfile1 = # convert into lower case
    wordfile1 = # split the wordfile
    wordfile1 = [ps."your answer here"(word) for word in wordfile1 if not word in set(stopwor
    wordfile1 = # join the wordfile with single blankspaces
    listofsent2.append(wordfile1)
```

```
# Create vectors of the above words using TF-IDF Vectorizer
# your code here
```

N-Grams

N-grams of texts are extensively used in text mining and natural language processing tasks. They are basically a set of co-occurring words within a given window and when computing the n-grams you typically move one word forward.



Documtation: https://en.wikipedia.org/wiki/N-gram

```
YouTubeVideo('E_mN90TYnlg', width=600, height=300)
```


```
# fill in the missing places for the code block
import nltk
from nltk.util import "your answer here"
samplText='CloudyMl is an EduTech Company'
NGRAMS="your answer here"(sequence=nltk."your answer here", n=1)
for grams in NGRAMS:
    print(grams)
```

```
('CloudyMl',)
('is',)
('an',)
('EduTech',)
('Company',)
```

```
samplText='CloudyMl is an EduTech Company'
# create ngrams with n=2
```

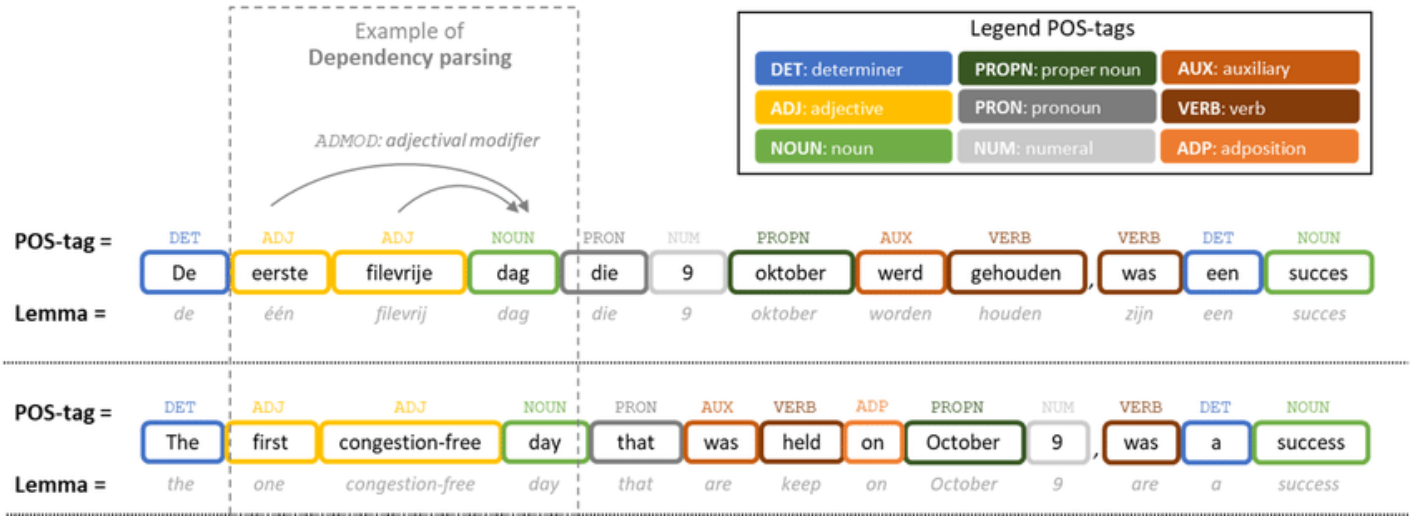
```
('CloudyMl', 'is')
('is', 'an')
('an', 'EduTech')
('EduTech', 'Company')
```

```
samplText='CloudyMl is an EduTech Company'
# create ngrams with n=3
```

```
('CloudyMl', 'is', 'an')
('is', 'an', 'EduTech')
('an', 'EduTech', 'Company')
```

Parts of speech(POS)

It is a process of converting a sentence to forms – list of words, list of tuples (where each tuple is having a form (word, tag)). The tag in case of is a part-of-speech tag, and signifies whether the word is a noun, adjective, verb, and so on.



Documentation:https://www.nltk.org/book/ch05.html

```
YouTubeVideo('EU18BuTvKmA?t=357', width=600, height=300)
```


```
# Parts of Speech.

# Universal Part-of-Speech Tagset

# Tag|Meaning|English Examples
# ADJ|adjective|new, good, high, special, big, local
```

```
# ADP|adposition|on, of, at, with, by, into, under
# ADV|adverb|really, already, still, early, now
# CONJ|conjunction|and, or, but, if, while, although
# DET|determiner, article|the, a, some, most, every, no, which
# NOUN|noun|year, home, costs, time, Africa
# NUM|numeral|twenty-four, fourth, 1991, 14:24
# PRT|particle|at, on, out, over per, that, up, with
# PRON|pronoun|he, their, her, its, my, I, us
# VERB|verb|is, say, told, given, playing, would
# .|punctuation marks|. , ; !


text = 'My name is CloudyML. I would to become a Data Scientist.'


# Importing Packages
# import nltk,word_tokenize and download averaged_perceptron_tagger,universal_tagset
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
True
```

```
# tokenize the given text into words



# implement parts of speech tagging using pos_tag



# print the tagged words
```

```
[('My', 'PRON'), ('name', 'NOUN'), ('is', 'VERB'), ('CloudyML', 'NOUN'), ('.', '.'), ('I
```

Great job!! You have come to the end of this practice sheet.
Treat yourself for this :))

Do Fill the feedback form !! [Feedback Form link]
You may head on to the next assignment.