Text Vectorization: How and why???

Why: Machine learning models don't understand text or image data directly as humans do

How: Convert the text data into numerical data, we need some smart ways which are known as vectorization, or in the NLP world, it is known as Word embeddings.

Reference:

https://www.analyticsvidhya.com/blog/2021/06/part-5-step-by-step-guide-to-master-nlp-text-vectorization-approaches/

Document

A document is a single text data point.

For Example, a review of a particular product by the user.

Dog hates a cat. It loves to go out and play.

Corpus

- It a collection of all the documents present in our dataset.

Let’s consider the 2 documents shown below,

Doc1: Dog hates a cat. It loves to go out and play.

Doc2: Cat loves to play with a ball.

Corpus = “Dog hates a cat. It loves to go out and play. Cat loves to play with a ball.”

Feature

Every unique word in the corpus is considered as a feature.

Features: [‘and’, ‘ball’, ‘cat’, ‘dog’, ‘go’, ‘hates’, ‘it’, ‘loves’, ‘out’, ‘play’, ‘to’, ‘with’]

Different types of Word Embeddings

1. Frequency-based or Statistical based Word Embedding
2. Prediction based Word Embedding

One-Hot Encoding (OHE)

Represent each unique word in vocabulary by setting a unique token with value 1 and rest 0 at other positions in the vector.

Example: I am teaching NLP in Python

Dictionary: [‘I’, ’am’, ’teaching’,’ NLP’,’ in’, ’Python’]

Vector for NLP: [0,0,0,1,0,0]

Vector for Python: [0,0,0,0,0,1]

Disadvantage

- Size of the vector is equal to the count of unique words in the vocabulary.
- Do not capture the relationships between different words. Therefore, it does not convey information about the context.

Count Vectorizer

- **1.** It is one of the simplest ways of doing text vectorization.
- **2.** It creates a document term matrix, which is a set of dummy variables that indicates if a particular word appears in the document.
- **3.** Count vectorizer will fit and learn the word vocabulary and try to create a document term matrix in which the individual cells denote the frequency of that word in a particular document, which is also known as term frequency, and the columns are dedicated to each word in the corpus.

Matrix Formulation

- Consider a Corpus C containing D documents {d1,d2…..dD} from which we extract N unique tokens.
- Now, the dictionary consists of these N tokens, and the size of the Count Vector matrix M formed is given by D X N.
- **Example:**

- Document-1: He is a smart boy. She is also smart.
- Document-2: Chirag is a smart person.

Unique Words: [‘He’, ’She’, ’smart’, ’boy’, ’Chirag’, ’person’]

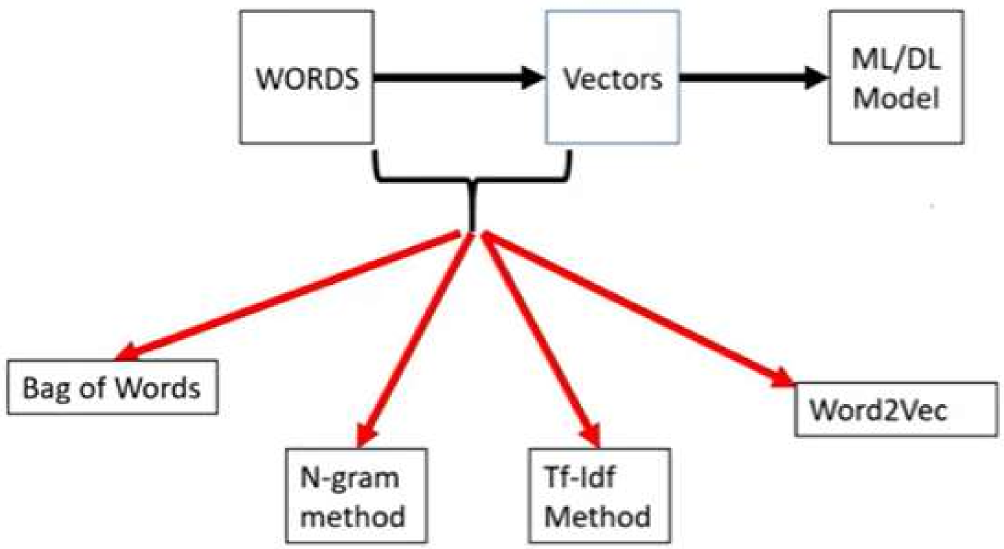
Here, D=2, N=6

So, the count matrix M of size 2 X 6 will be represented as –

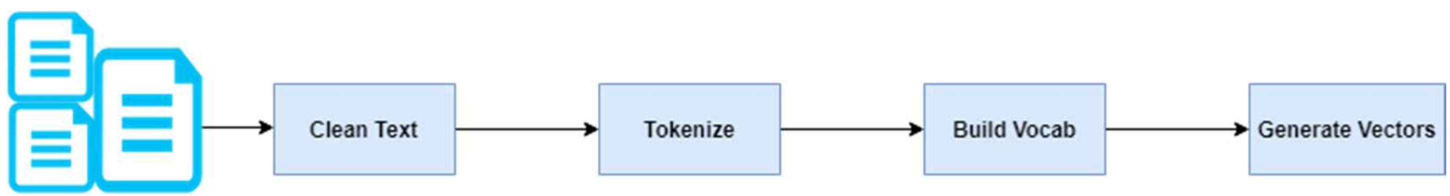
	He	She	smart	boy	Chirag	person
D1	1	1	2	1	0	0
D2	0	0	1	0	1	1

Vector for ‘smart’ is [2,1],
Vector for ‘Chirag’ is [0, 1], and so on.

Techniques for word vectorization



Preprocessing before vectorization

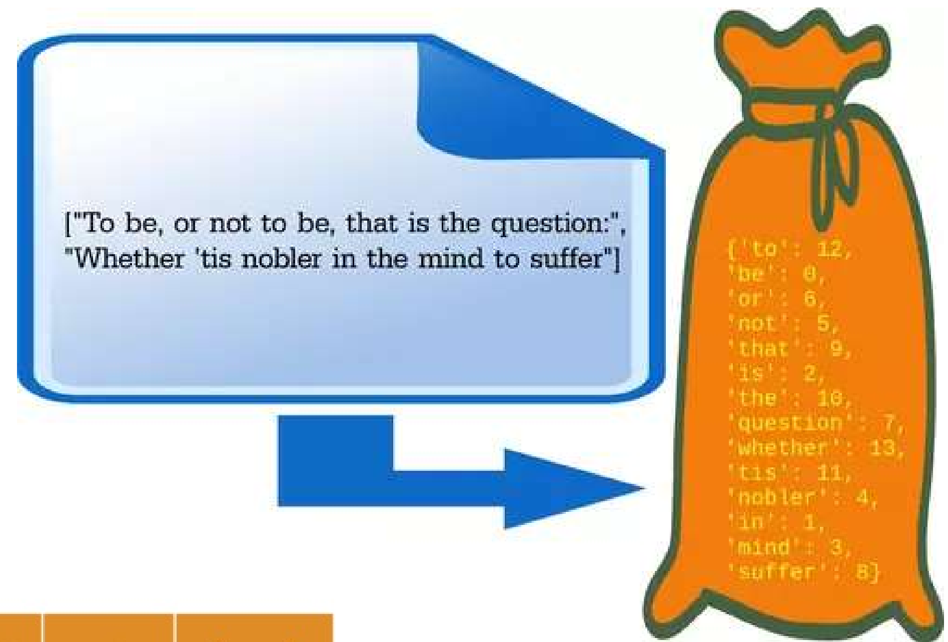


Bag of Words

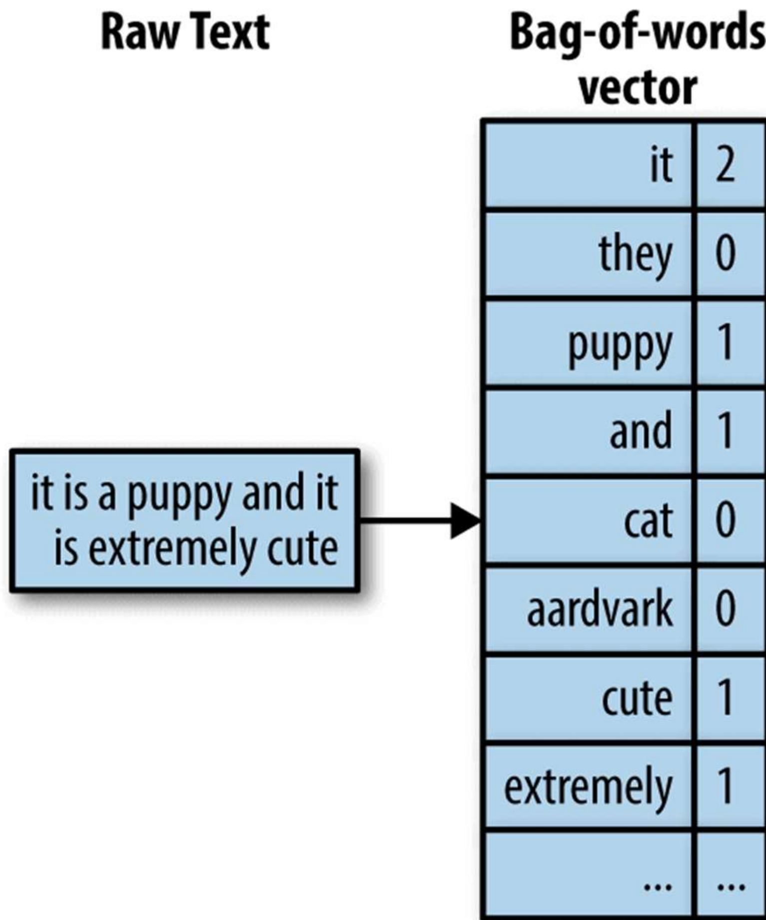
- It’s an algorithm that transforms the text into fixed-length vectors.
- Why call “Bag-Of-Words”????

Represents the sentence as a bag of terms.

Doesn’t take into account the order and the structure of the words.



	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


Raw Text
Bag-of-words vector
it is a puppy and it is extremely cute
it
2
they
0
puppy
1
and
1
cat
0
aardvark
0
cute
1
extremely
1
...
...

Disadvantage

- Weightage to unimportant words.
- No sequence preserved
- No method to preserve semantic meaning of sentence.
- Large dimension of input matrix.

Term frequency-inverse document frequency (TF-IDF)

- The problem with that is that it treats all words equally. As a result, it cannot distinguish very common words or rare words. So, to solve this problem.
- TF-IDF comes into the picture!
- Gives a measure that takes the importance of a word into consideration depending on how frequently it occurs in a document and a corpus.

TF-IDF (Term Frequency)

TFIDF works by proportionally increasing the number of times a word appears in the document but is counterbalanced by the number of documents in which it is present

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

Document 1: Text processing is necessary.

Document 2: Text processing is necessary and important.

Word	**TF**		**IDF**	**TFIDF**	
	Doc 1	**Doc 2**		**Doc 1**	**Doc 2**
Text	1/4	1/6	log (2/2) = 0	0	0
Processing	1/4	1/6	log (2/2) =0	0	0
Is	1/4	1/6	log (2/2) =0	0	0
Necessary	1/4	1/6	log (2/2) =0	0	0
And	0/4	1/6	log (2/1) =0.3	0	0.05
Important	0/4	1/6	log (2/1) =0.3	0	0.05

N-grams Vectorization

1. Similar to the count vectorization technique.

2. Document term matrix is generated, and each cell represents the count.

3. The columns represent all columns of adjacent words of length n.

4. Count vectorization is a special case of N-Gram where n=1.

5. N-grams consider the sequence of n words in the text; where n is (1,2,3..) like 1-gram, 2-gram. for token pair. Unlike BOW, it maintains word order.

Example

"I am studying NLP" has four words

- if n=2, i.e bigram, then the columns would be

  ["I am", "am reading", 'studying NLP"]

- if n=3, i.e trigram, then the columns would be

  ["I am studying", "am studying NLP"]

- if n=4,i.e four-gram, then the column would be

  ['"I am studying NLP"]

Example contd..

This is Big Data AI Book

Uni-Gram	This	Is	Big	Data	AI	Book
Bi-Gram	This is	Is Big	Big Data	Data AI	AI Book	
Tri-Gram	This is Big	Is Big Data	Big Data AI	Data AI Book		

Reference: https://www.analyticsvidhya.com/blog/2021/06/part-5-step-by-step-guide-to-master-nlp-text-vectorization-approaches/

Disadvantages of N-Grams

- **1.** It has too many features.
- **2.** Due to too many features, the feature set becomes too sparse and is computationally expensive.
- **3.** Choose the optimal value of N is not that easy task.

Summary

- **1.** Similar to the count vectorization method, in the TF-IDF method, a document term matrix is generated and each column represents an individual unique word.
- **2.** The difference in the TF-IDF method is that each cell doesn’t indicate the term frequency, but contains a weight value that signifies how important a word is for an individual text message or document

- This method is based on the frequency method but it is different from the count vectorization in the sense that it takes into considerations not just the occurrence of a word in a single document but in the entire corpus.

- **4.** TF-IDF gives more weight to less frequently occurring events and less weight to expected events