**Extracting the frequency of terms using the Bag of Words model**

One of the main goals of text analysis with the Bag of Words model is to convert text into a numerical form so that we can use machine learning on it. Let's consider text documents that contain many millions of words. In order to analyze these documents, we need to extract the text and convert it into a form of numerical representation.

Machine learning algorithms need numerical data to work with so that they can analyze the data and extract meaningful information. This is where the Bag of Words model comes in. This model extracts vocabulary from all the words in the documents and builds a model using a document-term matrix. This allows us to represent every document as a *bag of words*. We just keep track of word counts and disregard the grammatical details and the word order.

Let's see what a document-term matrix is all about. A document-term matrix is basically a table that gives us counts of various words that occur in a document. So, a text document can be represented as a weighted combination of various words. We can set thresholds and choose words that are more meaningful. In a way, we are building a histogram of all the words in the document that will be used as a feature vector. This feature vector is used for text classification.

Consider the following sentences:

* Sentence 1: The children are playing in the hall
* Sentence 2: The hall has a lot of space
* Sentence 3: Lots of children like playing in an open space

If you consider all three sentences, we have the following 14 unique words:

* the
* children
* are
* playing
* in
* hall
* has
* a
* lot
* of
* space
* like
* an
* open

Let's construct a histogram for each sentence by using the word count in each sentence. Each feature vector will be 14-dimensional because we have 14 unique words:

* Sentence 1: [2, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0]
* Sentence 2: [1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0]
* Sentence 3: [0, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1]

Now that we have extracted these features with the Bag of Words model, we can use machine learning algorithms to analyze this data.

**Let's see how to build a Bag of Words model in NLTK. Create a new Python file and import the following packages**:

import numpy as np

from sklearn.feature\_extraction.text import CountVectorizer

from nltk.corpus import brown

from text\_chunker import chunker

**Read the input data from the Brown corpus. We will use 5,400 words. Feel free to try it with as many words as you want:**

*# Read the data from the Brown corpus*

input\_data = ' '.join(brown.words()[:5400])

Define the number of words in each chunk:

*# Number of words in each chunk*

chunk\_size = 800

Divide the input text into chunks:

text\_chunks = chunker(input\_data, chunk\_size)

Convert the chunks into dictionary items:

# Convert to dict items

chunks = []

for count, chunk in enumerate(text\_chunks):

d = {'index': count, 'text': chunk}

chunks.append(d)

Extract the document term matrix where we get the count of each word. We will achieve this using the CountVectorizer method, which takes two input parameters. The first parameter is the minimum document frequency, and the second parameter is the maximum document frequency. The frequency refers to the number of occurrences of a word in the text:

# Extract the document term matrix

count\_vectorizer = CountVectorizer(min\_df=7, max\_df=20)

document\_term\_matrix = count\_vectorizer.fit\_transform([chunk['text'] for chunk in chunks])

Extract the vocabulary with the Bag of Words model and display it. The vocabulary refers to the list of distinct words that were extracted in the previous step:

*# Extract the vocabulary and display it*

vocabulary = np.array(count\_vectorizer.get\_feature\_names())

print("\nVocabulary:\n", vocabulary)

Generate the names for display:

# Generate names for chunks

chunk\_names = []

for i in range(len(text\_chunks)):

chunk\_names.append('Chunk-' + str(i+1))

Print the document-term matrix:

*# Print the document term matrix*

print("\nDocument term matrix:")

formatted\_text = '{:>12}' \* (len(chunk\_names) + 1)

print('\n', formatted\_text.format('Word', \*chunk\_names), '\n')

for word, item in zip(vocabulary, document\_term\_matrix.T):

*# 'item' is a 'csr\_matrix' data structure*

output = [word] + [str(freq) for freq in item.data]

print(formatted\_text.format(\*output))

The full code is given in the file bag\_of\_words.py. If you run the code, you will get the following output:

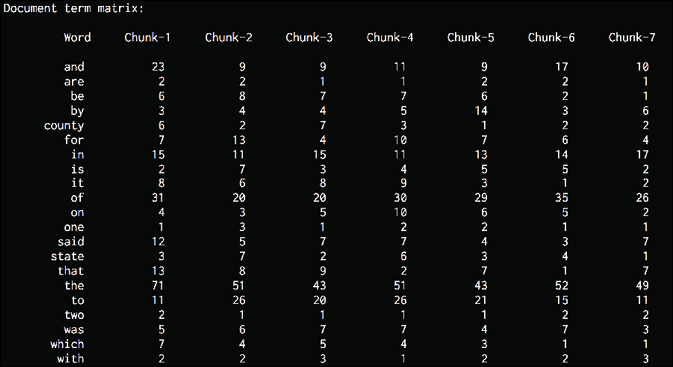


Figure 5: Document term matrix output

All the words can be seen in the Bag of Words model document-term matrix along with the corresponding counts in each chunk.

Now that we have done a count of the words, we can build on this and start making some predictions based on the frequency of words.