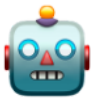

Module 3 Lab 2 - Using KNN for Text Classification

Created	@July 31, 2025 9:25 AM
Class	IIITH
Event	

https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/

"Lemmatizing" and "stemming" are both techniques used in natural language processing to reduce words to a base or root form. This is done to help group together different variations of a word so they can be analyzed as a single item.

Here's the difference:

- **Stemming:** This is a more basic process. It chops off the ends of words to get to a root. For example, the stemmer might reduce "running," "runs," and "runner" to "runn." The resulting stem might not be a real word. It's faster but can be less accurate because it doesn't consider the meaning or context of the word.
- **Lemmatization:** This is a more sophisticated process. It reduces words to their base or dictionary form, called the lemma. For example, the lemmatizer would reduce "running," "runs," and "ran" to the actual word "run." It considers the word's part of speech and context to find the correct base form. It's generally more accurate than stemming but also more computationally expensive.

In essence, both techniques aim to standardize words, but lemmatization is typically more precise as it results in a valid word, while stemming might produce a non-word root. The choice between them depends on the specific task and the desired balance between speed and accuracy.

Let's say we have the following two simple sentences as our training data:

Sentence 1: "The cat sat on the mat." Sentence 2: "The dog sat on the log."

And our test data is:

Sentence 3: "A cat sat on the log."

Here's how the `createBagOfWords` function would process this (assuming `remove_stopwords=True`, `lemmatize=False`, `stemmer=False` for simplicity in this example):

First, the `cleanText` function would clean the sentences. With stop words removed, the cleaned sentences might look something like:

Sentence 1: "cat sat mat" Sentence 2: "dog sat log" Sentence 3: "cat sat log"

Next, the `CountVectorizer` would learn the vocabulary from the training data (sentences 1 and 2). The unique words (tokens) are: "cat", "sat", "mat", "dog", "log".

Then, it creates the Bag-of-Words representation for the training data. Each row is a sentence, and each column is a word from the vocabulary:

	cat	sat	mat	dog	log
Sent 1	1	1	1	0	0
Sent 2	0	1	0	1	1

Finally, it creates the Bag-of-Words representation for the test data (sentence 3) using the *same* vocabulary learned from the training data:

	cat	sat	mat	dog	log
Sent 3	1	1	0	0	1

The function would return the two matrices shown above as `bag_of_words_train` and `bag_of_words_test`. These numerical matrices are what machine learning models use to understand and classify the text.