LVC 2: Text Preprocessing
Working with Text Data

Agenda

- Common Terminologies - NLP
- Text Data and the Need for NLP
- Challenges with Working with Text Data
- Text Preprocessing - Intro & Methods
- Word Vectorization - BoW & TF-IDF
- The Classical ML-NLP Pipeline

Common Terminologies

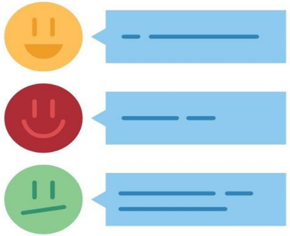
Documents Corpus Vocabulary Out-of-Vocabulary

In NLP, **a text body is referred to as a Document**. In other words, it is a collection of objects of the **text sequence** type, **known as a "string" in Python**.

Example: A document can be a single string or anything such as:

An Email

A News Article



A Movie Review

Common Terminologies

Documents | **Corpus** | Vocabulary | Out-of-Vocabulary

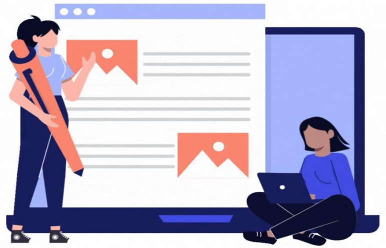
In NLP, a **corpus** (*plural* ***corpora***) is **a large set of text data that all NLP tasks depend upon**.

The corpus can consist of **a single document or a bunch of documents.** Such collections may be formed of **a single language of texts, or can span multiple languages.**

Example: A corpus can include:

All the articles of a news paper

Blogs from blogging sites



Threads of a discussion forum

Common Terminologies

Documents Corpus **Vocabulary** Out-of-Vocabulary

Vocabulary is the **set of distinct words that are used in a language**. **In NLP, vocabulary** refers to the **distinct words in a selected corpus.**

Example: Let the corpus be:

‘Bob ate apples. Fred ate apples. Bob and Fred ate pears.’

Then the vocabulary would be:

‘Bob’,**’ate’**, **‘apples’**, **‘Fred’**, **‘and’**, **‘pears’**

Common Terminologies

Documents Corpus Vocabulary **Out-of-Vocabulary**

In NLP, **any word** in a document **which is not found in the relevant corpus vocabulary** is considered **Out of Vocabulary.**

Example: Let the corpus be:

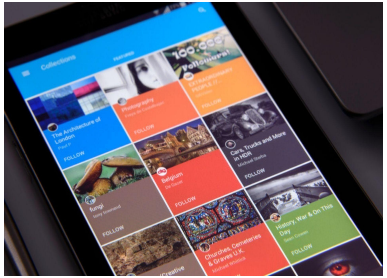
‘Bob ate apples. Fred ate apples. Bob and Fred ate pears.’

Here, the word **‘eat’** would represent an **Out of Vocabulary** word. Any word that is not in the corpus of the NLP task is considered an Out of Vocabulary word.

Information Exchange - The Purpose of Text Data

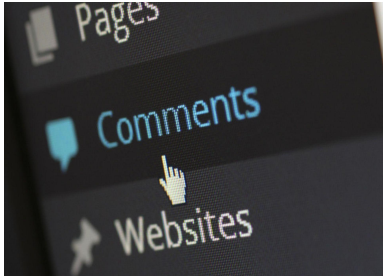
- All of us engage very frequently in information generation and exchange. This exchange is done in multiple ways such as reading, writing, watching, speaking, listening etc.
- In addition, **this information exchange happens everyday and everywhere** in our day-to-day lives.
- Some of the examples are:

Meetings and catch-up calls



Reading the news

Prescription from a doctor



Exchanging ideas on social media

- There are many other such examples of information exchange, like paying for items purchased at the point of sales, checking in for a flight, signing and providing details during checking into a hotel, speaking to a friend, so on and so forth.

Text Data

- In all these instances of information generation and exchange, **there is usually some form of communication taking place between human beings.**

- During this information exchange, an enormous amount of data through natural language (in either written or spoken form) is being generated by an individual or an organization.

- The written, **recorded form of this language-based communication** is what’s referred to as **text**. It is often made up of documents that contain words, phrases or even paragraphs of content.

- Text, or more generally natural language, is the **most important form of communication between human beings**, and as such, the recorded volume of text data we currently have access to is massive and extremely important for us to analyze and leverage across business and industry, with more than 20 billion texts and 500 million tweets being sent every single day.

- Text data is a potential source of information about the likes, dislikes, preferences, trends, risks, and expectations of users in various contexts. Not only do we have all this at our disposal, it also gives us a good idea about how language is used to communicate in the real world, through a whole body of knowledge in the form of text.

The Need for Natural Language Processing

- **Text data is different**, because it cannot directly be input into **Machine Learning** and **Deep Learning** models like numerical, tabular data. ***Text data requires preprocessing and vectorization*** before it can be analyzed and mined for insights.

- However it is worth noting that **even though text data is unstructured, it is sequential in nature**. This is obvious from the fact that changing or reversing the order of characters or words in a sentence, changes its meaning or renders it meaningless. For example, consider the sentence:

  "The cat and the dog sat on the wall."

- Reversing the sentence to "wall the on sat dog the and cat the" or just swapping two words to form "The cat and the wall sat on the dog" can entirely change the meaning of the sentence.

- This should give a sense of the need for a unique class of models capable of making predictions on text data, which is a different data modality from tabular data, and this is what establishes the need for NLP as a separate domain within Data Science and Artificial Intelligence

Challenges with Text Data

- **Contextual words, Phrases and Homonyms - ** The same **words and phrases can have different meanings** according to the context of a sentence and many words. For example:
  - Let's go hang out by the **pool**.
  - Let's go play some **pool** tomorrow.

- **Irony, Sarcasm & Ambiguity -** Humans generally use **words and phrases that may be positive or negative according to the dictionary, but actually signify the opposite meaning**, whereas ambiguity refers to sentences and phrases that are likely to be two or more possible interpretations. For example:
  - When the weather is bad and someone says, "What lovely weather we have!"
  - When someone is moving too slowly: "Could you please move more slowly?"

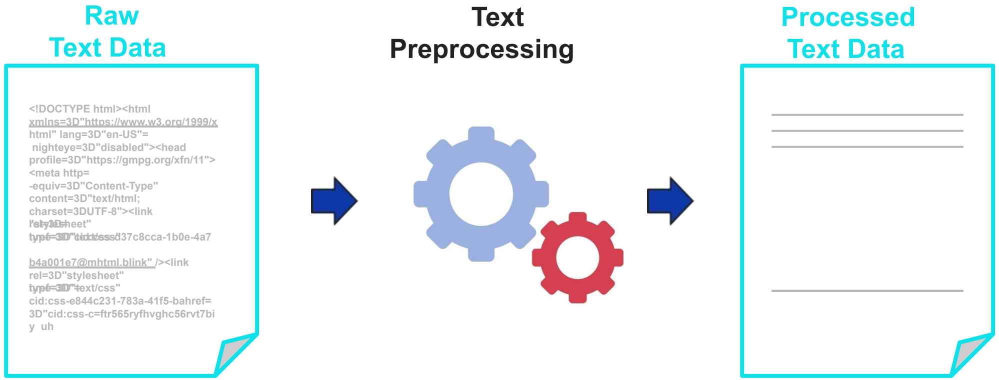
- **Colloquialisms and Slang -** Informal phrases, expressions, idioms, and culture-specific lingo present numerous issues for NLP models. For example, some people use the phrases like:
  - “Hard to swallow” when they mean difficult to believe.
  - “Knee-jerk reaction” when they mean a quick or automatic response.

- **Domain-specific Language - Different businesses and industries often use very different languages for their activities and operations.** For example, an NLP processing model needed for healthcare would be very different than the one used to process legal documents.

Introduction to Text Preprocessing

- **The process of transforming raw and unclean text data into a form that is analyzable for the model is known as Text Preprocessing.**

- For every NLP task, a lot of preprocessing goes on behind the scenes so that a model can understand and interpret text data. If we train our model on unstructured data that hasn't been preprocessed, **the model can miss out on learning important information.**



```
<!DOCTYPE html><html
xmlns=3D"https://www.w3.org/1999/x
html" lang=3D"en-US"=
 nighteye=3D"disabled"><head
profile=3D"https://gmpg.org/xfn/11">
<meta http=
-equiv=3D"Content-Type"
content=3D"text/html;
charset=3DUTF-8"><link
rel=3D"stylesheet"
type=3D"text/css" href=3D"cid:css-d37c8cca-1b0e-4a7

b4a001e7@mhtml.blink" /><link
rel=3D"stylesheet"
type=3D"text/css" href=3D"
cid:css-e844c231-783a-41f5-bahref=
3D"cid:css-c=ftr565ryfhvghc56rvt7bi
y  uh
```

- **What happens if we do not preprocess text data?** When **text data is not suitably processed**, it tends to be **noisy in character** and the **quality of the model built on top of it gets impacted.**

Common Text Preprocessing Methods

- A few of the common Text Preprocessing methods we will explore this module are:

1 **Treating accented and special characters**

2 **Lowercasing**

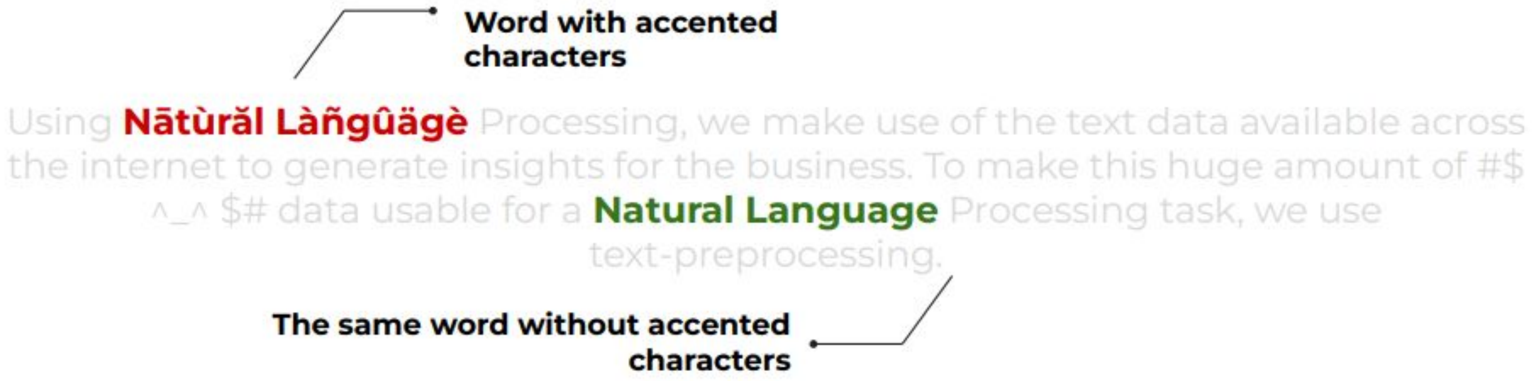
3 **Stop word Removal**

4 **Tokenization**

5 **Stemming & Lemmatization**

Accented & Special Characters

- In English text preprocessing, accented characters have generally been removed when the Natural Language applications being developed have been simple. This would of course be different for other languages that rely more heavily on accented characters.



Word with accented characters

Using Nātùrăl Làñgûäġè Processing, we make use of the text data available across the internet to generate insights for the business. To make this huge amount of #$ ^_^ $# data usable for a Natural Language Processing task, we use text-preprocessing.

The same word without accented characters

- Special Characters have also generally been removed from text in classical NLP. Anything that is **not an alphabetic or numeric character** can be considered a Special Character.

  Some examples are **! ,#, $, &, @**

Lowercasing & Stop word Removal

- **Lowercasing simply converts all words into lowercase** to ensure that **differently-cased / capitalized occurrences of the same word in different cases are still treated as the same word**.

- The idea behind **Stop word Removal**, on the other hand, is that **by eliminating low-information parts of the text, we can concentrate on the keywords that give the text most of its meaning**.

- However, **the incorrect removal of stop words, especially in small sentences, can even alter the meaning of our text**.

- For example, consider the below text:

‘The movie was **not** bad’

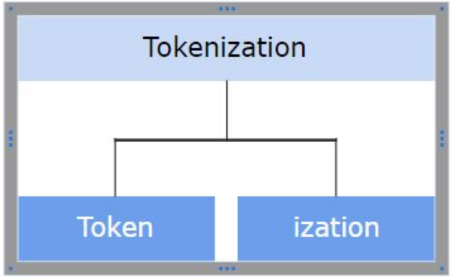
- In this example - the word **not** cannot be considered a Stop word for removal - removing it would change the meaning of the entire sentence.

Tokenization

- **Tokenization** is simply **the process of breaking a stream of raw text into small chunks of sub-words, words or sentences known as tokens.**

- In Classical NLP, Tokenization was generally done either at the Word Level (**Word Tokenization**) or at the Sentence Level (**Sentence Tokenization**).

- Word Tokenization involves splitting text into its constituent words, and Sentence Tokenization does that into sentences.



What the industry has found in recent years though, is that **Sub-word Tokenization, splitting text into sub-words, is in fact the main Text Preprocessing technique that works Large Language Models.** Sub-word Tokens are in fact the smallest atomic unit of understanding and prediction for LLMs - they understand text as a sequence of sub-word tokens, and they are repeatedly only predicting the next sub-word token until they complete their sentences.

Stemming & Lemmatization

- **Stemming and Lemmatization** are two Word Standardization techniques that attempt to reduce a word to some sort of base / root form to standardize different forms of the same word that may exist in the same corpus. They merely accomplish this in slightly different ways.

- **Stemming is typically rule-based**, meaning a word is analyzed and run through a series of conditions to determine how to cut it down to its base form or stem. **Lemmatization**, on the other hand, **minimizes words to a canonical / dictionary relevant base form.**

Word	Lemmatization Output	Stemming Output
Changes	**Change**	**Changes**
Changing	**Change**	**Changing**
Multiplied	**Multiply**	**Multiplied**
Multiplier	**Multiply**	**Multiplier**

As we see in the table on the left, t**he root word outputs from Stemming & Lemmatization can differ for the same word.**

Word Vectorization

- Machines, as we all know, **cannot really understand text as input**.

- So in order to perform Machine Learning on text, even after Preprocessing, we need to **convert text into a numerical format that machines can understand, to find patterns and make predictions. This is referred to as Word Representation or Word Vectorization,** and it is the fundamental idea behind how Natural Language Processing and also Deep Learning / Generative AI methods are able to understand and make predictions on text data.

- Here, we shall talk about two of the classical count-based NLP techniques that were first used to vectorize text:

1 **Bag of Words (BoW)**

2 **TF-IDF**

Bag of Words (BoW)

- **Bag of Words**, the simplest Text Vectorization technique, is simply a modified version of One-hot Encoding, in which, rather than storing whether a word in the dictionary is present or absent, **we store the count of that word occurrence in the corpus.**

- As an example, for the sentence “I like cats, do you like cats?” - this table has the counts that would go into the vector representations.

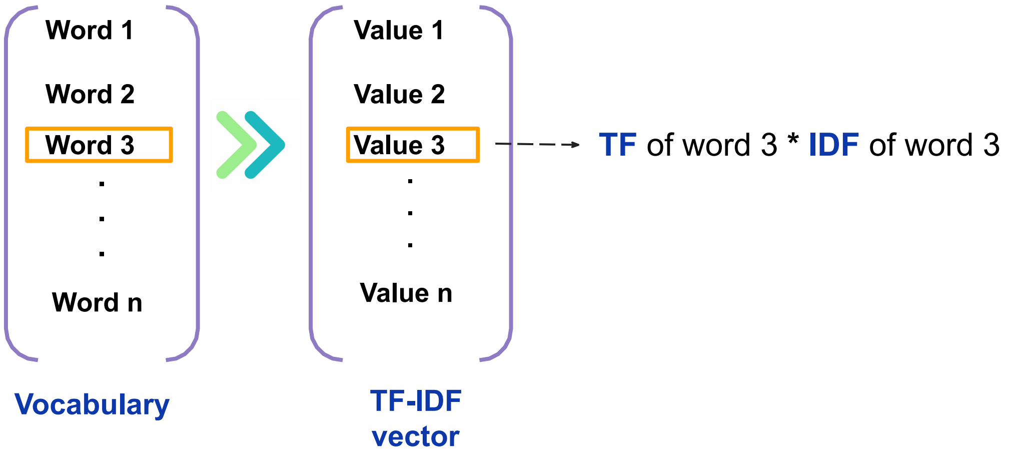
“ I like cats, do you like cats? “

Word	I	like	cats	do	you
Count	1	2	2	1	1

Term-Frequency Inverse Document Frequency (TF-IDF)

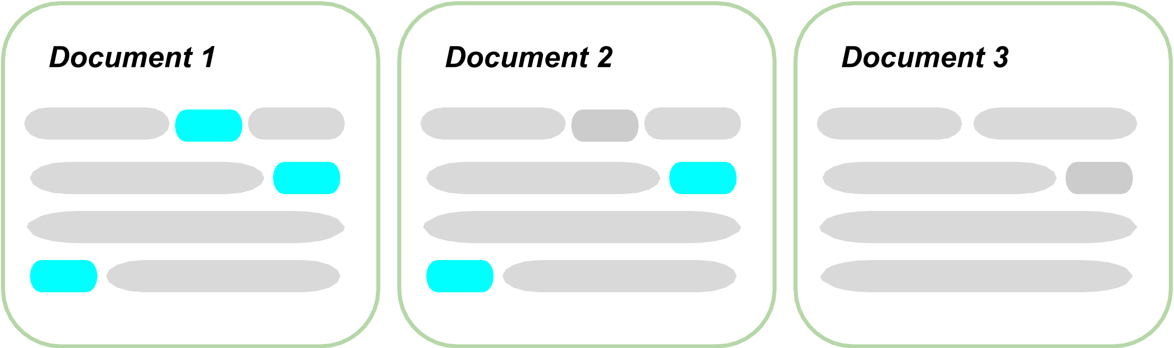
- While Bag of Words simply uses the count as features of the vector, **TF-IDF** uses statistical measures to **evaluate how important a word is in a corpus.**

- The **value** in the resulting **vector** corresponding to each word is the **product of TF (Term Frequency) and IDF (Inverse Document Frequency).**



Term-Frequency Inverse Document Frequency (TF-IDF)

- However, the **Inverse Document Frequency (IDF)** is the **logarithm of the ratio of the total number of documents versus the number of documents in which the word “X” appears.**



# Doc = 3

# Doc containing X = 2

IDF of X = log (3/2)

- So the TF-IDF for the word “X” in Document 1, would be the product of 3/120 and log (3/2). There is essentially a TF-IDF score for every word in every document.

Term-Frequency Inverse Document Frequency (TF-IDF)

So how does TF-IDF work?

- **The IDF** part essentially works as a dampening factor to **reduce the importance of terms that are common** to a lot of documents.

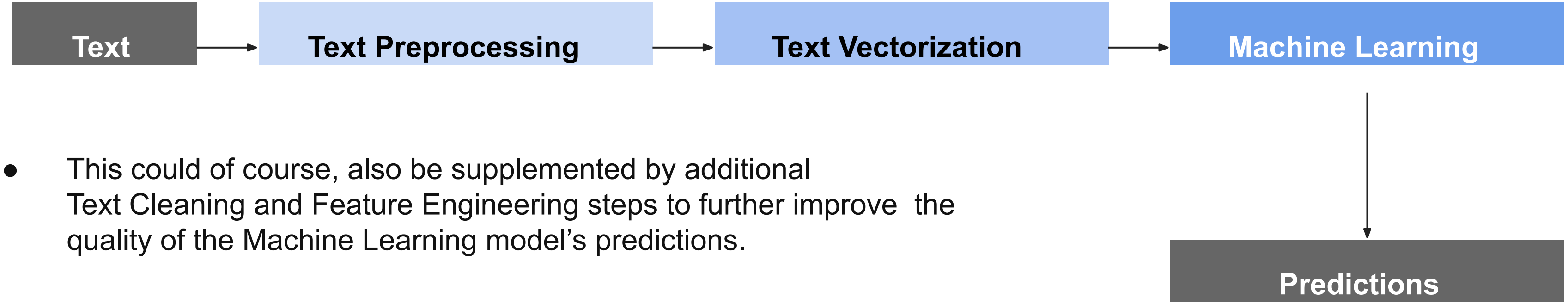
  For example, terms like “**about**” or “**but**” would occur **a lot in an article**. But that **does not mean that they contribute a lot to its meaning.** This is what **IDF is used for**.

- On the other hand, **when a keyword appears only in a small number of documents**, it is deemed **more relevant to the documents in which it appears**. TF helps us obtain this information.

- In this way, **TF-IDF attempts to give higher relevance scores to words that occur in fewer documents within the corpus.**

The Classical Machine Learning NLP Pipeline

- So in a nutshell, the classical Machine Learning NLP pipeline looks like the following:



- This could of course, also be supplemented by additional Text Cleaning and Feature Engineering steps to further improve the quality of the Machine Learning model's predictions.

The Need for NLP

Great Learning

Happy Learning !