**DAY 10**

**30.06.2023**

**Text Processing**

Text Processing is one of the most common tasks in many ML applications. Below are some examples of such applications.

* ***Language Translation:***

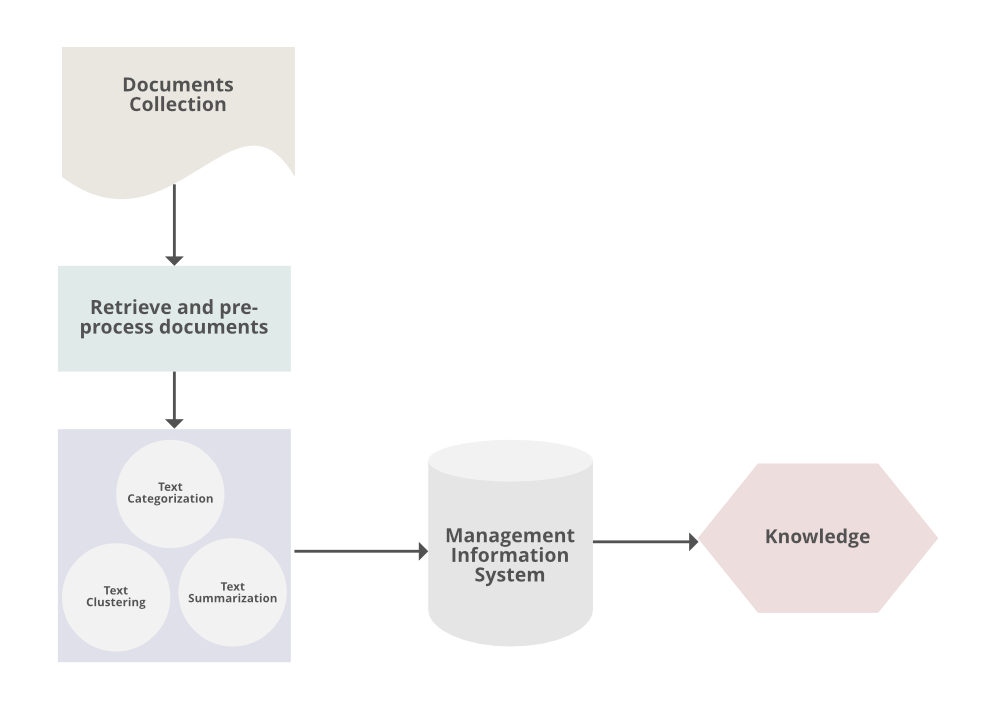
Translation of a sentence from one language to another.

* ***Spam Filtering:***

Detect unsolicited and unwanted email / messages.

* ***Sentiment Analysis:***

To determine, from a text corpus, whether the sentiment towards any topic or product etc. is positive, negative, or neutral.



These applications deal with huge amount of text to perform classification or translation and involves a lot of work on the back end. Transforming text into something an algorithm can digest is a complicated process.

***The steps involved in text processing***

* Data Preprocessing
* Feature Extraction
* Choosing ML Algorithms

**[i] Data Preprocessing**

**Natural Language Processing (NLP) is a branch of artificial intelligence that deals with the interactions between computers and human languages. NLP allows computers to understand and process human language in a way that is similar to how humans do. It has many practical applications in the business world, including language translation, sentiment analysis, etc.**

* ***Lemmatization:***

Another approach to remove inflection by determining the part of speech and utilizing detailed database of the language.

* ***Stemming:***

Words are reduced to a root by removing inflection through dropping unnecessary characters, usually a suffix.

* ***Removing stop words:***

Frequent words such as ”the”, ”is”, etc. that don’t have specific semantic.

* ***Tokenization:***

Convert sentences to words.

* ***Removing unnecessary punctuation, tags***

***Preprocessing operations using Python***

* [***NLTK***](http://www.nltk.org/)***:*** The Natural Language ToolKit is one of the best-known and most-used NLP libraries, useful for all sorts of tasks from t tokenization, stemming, tagging, parsing, and beyond.
* [***BeautifulSoup***](https://www.crummy.com/software/BeautifulSoup/bs4/doc/)***:*** Library for extracting data from HTML and XML documents.

**[ii] Feature Extraction**

In text processing, words of the text represent discrete, categorical features. The mapping from textual data to real valued vectors is called feature extraction. One of the simplest techniques to numerically represent text is **Bag of Words.**

***Bag of Words (BOW):***

The list of unique words in the text corpus called vocabulary. Then it can represent each sentence or document as a vector with each word represented as 1 for present and 0 for absent from the vocabulary.

The most popular approach is using the **Term Frequency-Inverse Document Frequency (TF-IDF)** technique.

* ***Term Frequency (TF) =*** Number of times term t appears in a document / Number of terms in the document.
* ***Inverse Document Frequency (IDF) =*** log(N/n), where, N is the number of documents and n is the number of documents a term t has appeared in.
* ***TF-IDF = TF \* IDF***

One of the major disadvantages of using BOW is that it discards word order thereby ignoring the context and in turn meaning of words in the document.

***Word Embedding:***

It is a representation of text where words that have the same meaning have a similar representation. In other words, it represents words in a coordinate system where related words, based on a corpus of relationships, are placed closer together. Work2Vec is the well-known model of word embedding.

## ****GloVe:****

## The Global Vectors for Word Representation, or [GloVe](http://doi.org/10.3115/v1/D14-1162" \t "_blank), algorithm is an extension to the word2vec method for efficiently learning word vectors. GloVe constructs an explicit word-context or word co-occurrence matrix using statistics across the whole text corpus. The result is a learning model that may result in generally better word embeddings.

**[iii] Choosing ML Algorithms**

There are various approaches to building ML models for various text-based applications depending on what is the problem space and data available.

***Classical ML approaches for spam filtering***

* ‘Naive Bayes’ or ‘Support Vector Machines’.
* Widely used ML approach.

Deep learning techniques are giving better results for NLP problems like sentiment analysis and language translation. Deep learning models are very slow to train and it has been seen that for simple text classification problems classical ML approaches as well give similar results with quicker training time.

**References**

<https://monkeylearn.com/blog/text-processing/>

<https://www.techopedia.com/definition/22541/text-processing>

<https://www.datarobot.com/blog/text-processing-what-why-and-how/>

<https://www.analyticsvidhya.com/blog/2021/09/essential-text-pre-processing-techniques-for-nlp/>