Text mining

Statistiques textuelles - LZML041

Séance 2 : Le Corpus

Chargé de cours : Loubna Serrar

ANNÉE UNIVERSITAIRE 2022-2023

Description du cours

- 12 séances de 2h pour **s'initier au Text mining**
- Un mix de théorie et de pratique avec **l'outil Python/Jupyterlab**
- **Évaluation en controle continu** avec des tests courts toutes les 3 séances et un travail final à rendre en mai (50/50)
- **Plan des premières séances:**

1. **Introduction Générale**
2. **What is a corpus ?**
3. **pre-processing techniques** (Cleaning? Segmentation, tokenization, part of speech tagging, syntactic parser, data annotation…)
4. **Text to numbers** (Bag of words, frequency, TF-IDF, n-gramm model…

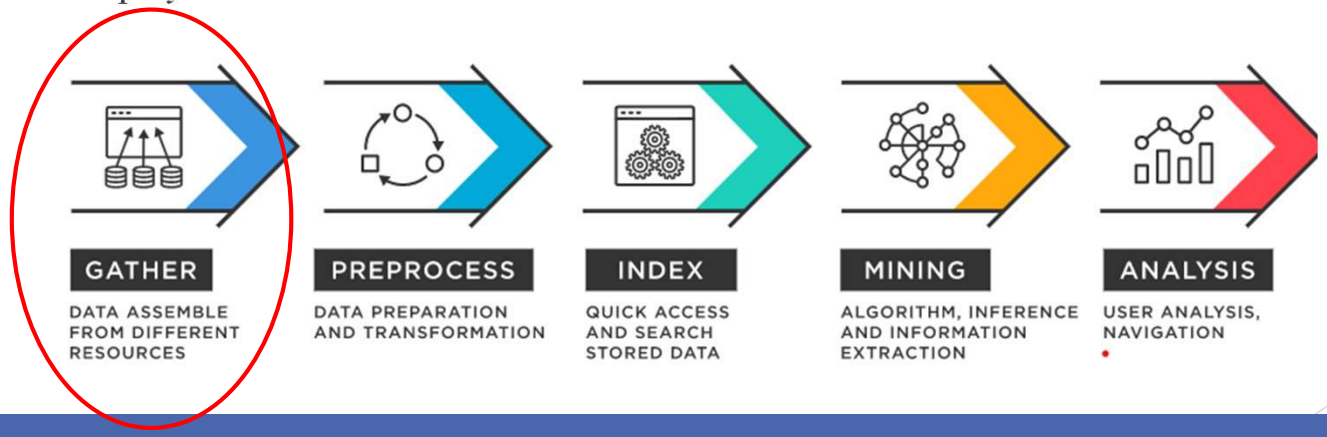
ChatGPT - https://chat.openai.com/chat

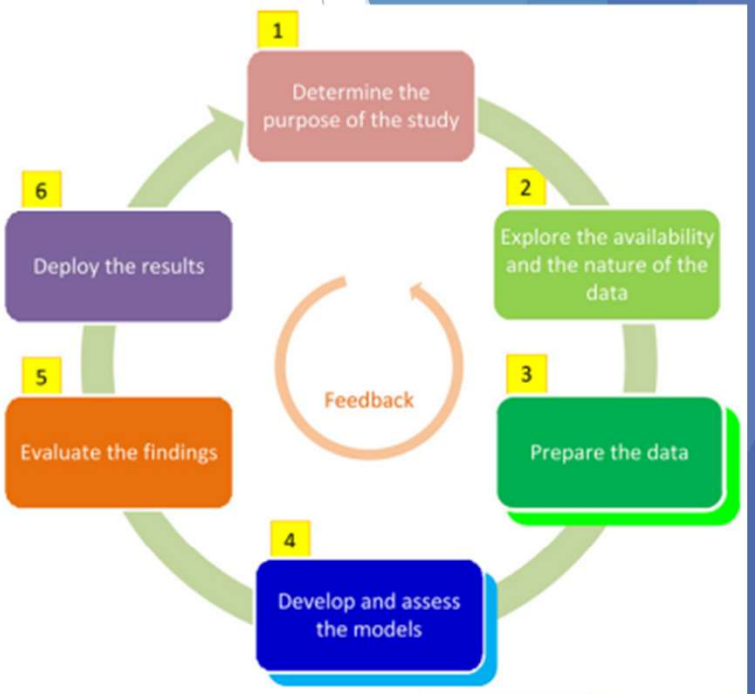
https://www.youtube.com/watch?v=8kIxMUrdLQw&ab_channel=Numerama

How does Text Mining work?

- The six phases that naturally describes the data science life cycle of a data mining project:

1. Determining the purpose of the study: what is the question we want to answer?
2. Data understanding: what data do we have / need?
3. Data preparation: how do we organize the data for modeling?
4. Modeling: what modeling techniques should we apply?
5. Evaluation: which model best meets the study objectives?
6. Deployment: how do users access the results?





What is a corpus?

- Corpora are **collections of related documents** that contain natural language.
- A corpus can vary in size, from tweets to books, but they contain text (and sometimes metadata) inside of thousands of documents.
- Documents can in turn be broken into:
  - **Paragraphs units of discourse** that generally each express a single idea
  - Paragraphs can be further broken down into **sentences, which are units of syntax**,
  - Sentences are made up of **words and punctuation**, the lexical units that indicate general meaning but are far more useful in combination.
  - Finally, words themselves are made up of **syllables, phonemes, affixes**, and characters, units that are only meaningful when combined into words

CHOOSE A DOMAIN SPECIFIC CORPUS

- Different domains use different language (vocabulary, acronyms, common phrases, etc.), so a corpus that is relatively pure in domain will be able to be analyzed and modeled better than one that contains documents from several different domains.

General

- WordNet® is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept.
- Google Blogger Corpus: Nearly 700,000 blog posts from blogger.com. The meat of the blogs contain commonly occurring English words, at least 200 of them in each entry.
- Enron Dataset: Over half a million anonymized emails from over 100 users. It's one of the few publically available collections of "real" emails available for study and training sets.
- SMS Spam Collection: Excellent dataset focused on spam. Nearly 6000 messages tagged as legitimate or spam messages with a useful subset extracted directly from Grumbletext.
- Recommender Systems Datasets: Datasets from a variety of sources, including fitness tracking, video games, song data, and social media. Labels include star ratings, time stamps, social networks, and images.

Specific:

- 20 Newsgroups: 20,000 documents from over 20 different newsgroups. The content covers a variety of topics with some closely related for reference. There are three versions, one in its original form, one with dates removed, and one with duplicates removed.
- The WikiQA Corpus: Contains question and sentence pairs. It's robust and compiled from Bing query logs. There are over 3000 questions and over 29,000 answer sentences with just under 1500 labeled as answer sentences.
- European Parliament Proceedings Parallel Corpus: Sentence pairs from Parliament proceedings. There are entries from 21 European languages including some less common entries for ML corpus.
- Jeopardy: Over 200,000 questions from the famed tv show. It includes category and value designations as well as other descriptors like question and answer fields and rounds.
- Legal Case Reports Dataset: Text summaries of legal cases. It contains wrapups of over 4000 legal cases and could be great for training for automatic text summarization.
- LibriSpeech: Nearly 1000 hours of speech in English taken from audiobook clips.

Litterature

- Project Gutenberg: Extensive collection of book texts. These are public domain and available in a variety of languages, spanning a long period of time.
- ABU : l'Association des Bibliophiles Universels ou ABU se propose de maintenir un corpus le plus vaste possible de textes numérisés en français du domaine public, représentatifs de la culture francophone.

Python Toolkit (NLTK)

- **The Natural Language Toolkit (NLTK)** is an open source library for building Python programs that work with human language data for applying in statistical natural language processing (NLP). Online NLKT guide here.
- It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

Language processing tasks and corresponding NLTK modules with examples of functionality

Language processing task	NLTK modules	Functionality
Accessing corpora	corpus	standardized interfaces to corpora and lexicons
String processing	tokenize, stem	tokenizers, sentence tokenizers, stemmers
Collocation discovery	collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	tag	n-gram, backoff, Brill, HMM, TnT
Machine learning	classify, cluster, tbl	decision tree, maximum entropy, naive Bayes, EM, k-means
Chunking	chunk	regular expression, n-gram, named-entity
Parsing	parse, ccg	chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	sem, inference	lambda calculus, first-order logic, model checking
Evaluation metrics	metrics	precision, recall, agreement coefficients
Probability and estimation	probability	frequency distributions, smoothed probability distributions
Applications	app, chat	graphical concordancer, parsers, WordNet browser, chatbots
Linguistic fieldwork	toolbox	manipulate data in SIL Toolbox format

Lets open Python / Jupyter !

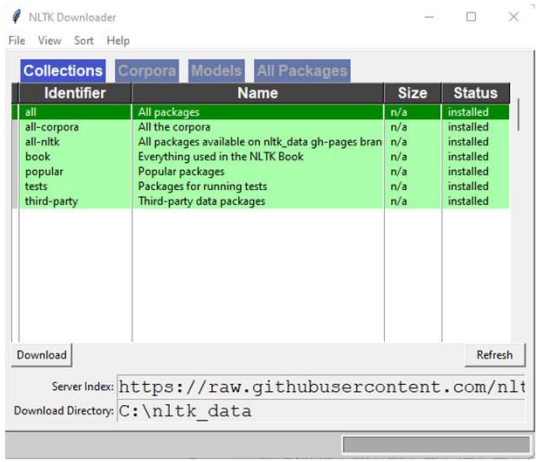
- **The Natural Language Toolkit (NLTK)** is an open source library for building Python programs that work with human language data for applying in statistical natural language processing (NLP). Online NLKT guide here.
- Installing **NLTK library :** do the following in your Jupyter notebook.

```
pip install nltk

import nltk

nltk.download()
```

- ✓ A new window should open, showing the NLTK Downloader. Click on the File menu and select Change Download Directory. For central installation, set this to C:\nltk_data (Windows), /usr/local/share/nltk_data (Mac), or /usr/share/nltk_data (Unix). Next, download all.




```
[1]: pip install numpy

Requirement already satisfied: numpy in d:\users\loubn\anaconda3\lib\site-packages (1.21.5)
Note: you may need to restart the kernel to use updated packages.

[2]: pip install nltk

Requirement already satisfied: nltk in d:\users\loubn\anaconda3\lib\site-packages (3.7)
Requirement already satisfied: tqdm in d:\users\loubn\anaconda3\lib\site-packages (from nltk) (4.64.1)
Requirement already satisfied: regex>=2021.8.3 in d:\users\loubn\anaconda3\lib\site-packages (from nltk) (2022.7.9)
Requirement already satisfied: joblib in d:\users\loubn\anaconda3\lib\site-packages (from nltk) (1.1.0)
Requirement already satisfied: click in d:\users\loubn\anaconda3\lib\site-packages (from nltk) (8.0.4)
Requirement already satisfied: colorama in d:\users\loubn\anaconda3\lib\site-packages (from click->nltk) (0.4.5)
Note: you may need to restart the kernel to use updated packages.

[*]: # First import the library NLKT
import nltk
nltk.download()

showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml

[2]: # Sample usage for wordnet
# WordNet is just a NLTK corpus reader, and can be imported like this:
from nltk.corpus import wordnet as wn

[3]: wn.langs()

[3]: dict_keys(['eng', 'als', 'arb', 'bul', 'cmn', 'dan', 'ell', 'fin', 'fra', 'heb', 'hrv', 'isl', 'ita', 'ita_iwn', 'jpn', 'cat', 'eus',

[6]: wn.synsets('chat', lang='fra')
```

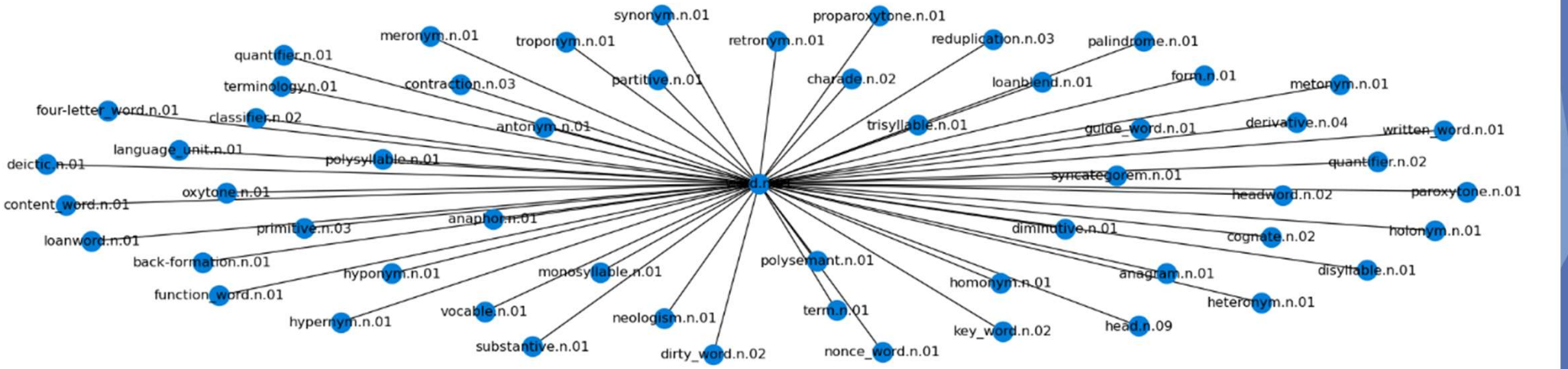
Lets do some Python !

Jupyterlab keyboard shortcuts

- Command Mode : press Esc to enable)
  - F : find and replace
  - Ctrl-Shift-F : open the command palette
  - Ctrl-Shift-P : open the command palette
  - Enter : enter edit mode
  - P : open the command palette
  - Shift-Enter : run cell, select below
  - Ctrl-Enter : run selected cells
  - Alt-Enter : run cell and insert below
  - **Y : change cell to code**
  - M : change cell to markdown
  - R : change cell to raw
  - 1 : change cell to heading 1
  - 2 : change cell to heading 2
  - 3 : change cell to heading 3
  - 4 : change cell to heading 4
  - 5 : change cell to heading 5
  - 6 : change cell to heading 6
  - K : select cell above
  - Up : select cell above
  - Down : select cell below
  - J : select cell below
  - Shift-K : extend selected cells above
  - Shift-Up : extend selected cells above
  - Shift-Down : extend selected cells below
  - Shift-J : extend selected cells below
  - Ctrl-A : select all cells
  - A : insert cell above
  - B : insert cell below
  - X : cut selected cells
  - C : copy selected cells
  - Shift-V : paste cells above
  - V : paste cells below
  - Z : undo cell deletion
  - D,D : delete selected cells
  - Shift-M : merge selected cells, or current cell with cell below if only one cell is selected
  - Ctrl-S : Save and Checkpoint
  - S : Save and Checkpoint
  - L : toggle line numbers
  - O : toggle output of selected cells
  - Shift-O : toggle output scrolling of selected cells
  - H : show keyboard shortcuts
  - I,I : interrupt the kernel
  - 0,0 : restart the kernel (with dialog)
  - Ctrl-V : Dialog for paste from system clipboard
  - Esc : close the pager
  - Q : close the pager
  - Shift-L : toggles line numbers in all cells, and persist the setting
  - Shift-Space : scroll notebook up
  - Space : scroll notebook down

Corpus example 1 : Wordnet

- WordNet, created by Princeton is a lexical database for English language. It is the part of the NLTK corpus.
- NLTK module includes the English WordNet **with 155 287 words and 117 659 synonym sets** that are logically related to each other.
- In WordNet, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms called **Synsets.** All the synsets are linked with the help of conceptual-semantic and lexical relations. Its structure makes it very useful for natural language processing (NLP).
- In information systems, WordNet is used for various purposes like word-sense disambiguation, information retrieval, automatic text classification and machine translation. One of the most important uses of WordNet is to find out the similarity among words.



Lets do some Python !

Ambiguity and Uncertainty in Language

- Lexical Ambiguity : The ambiguity of a single word is called lexical ambiguity. For example, treating the word **silver** as a noun, an adjective, or a verb.
- Syntactic Ambiguity : This kind of ambiguity occurs when a sentence is parsed in different ways. For example, the sentence “The man saw the girl with the telescope”. It is ambiguous whether the man saw the girl carrying a telescope or he saw her through his telescope.
- Semantic Ambiguity : This kind of ambiguity occurs when the meaning of the words themselves can be misinterpreted. In other words, semantic ambiguity happens when a sentence contains an ambiguous word or phrase. For example, the sentence “The car hit the pole while it was moving” is having semantic ambiguity because the interpretations can be “The car, while moving, hit the pole” and “The car hit the pole while the pole was moving”.
- Anaphoric Ambiguity : This kind of ambiguity arises due to the use of anaphora entities in discourse. For example, the horse ran up the hill. It was very steep. It soon got tired. Here, the anaphoric reference of “it” in two situations cause ambiguity.
- Pragmatic ambiguity : Such kind of ambiguity refers to the situation where the context of a phrase gives it multiple interpretations. In simple words, we can say that pragmatic ambiguity arises when the statement is not specific. For example, the sentence “I like you too” can have multiple interpretations like I like you (just like you like me), I like you (just like someone else dose).

Corpus example 2 : Guttenberg

- The Gutenberg dataset represents a corpus of over 15,000 book texts, their authors and titles, all available on : https://www.gutenberg.org/
- The text data itself can be downloaded using the gutenberg_download.py script, which will parse the metadata file, download the text data for each book and save the results as a csv file. The final csv file containing the book texts, the authors, the titles and the categories will have a size of around 5 GB. https://www.kaggle.com/datasets/mateibejan/15000-gutenberg-books
- But we will use another code to select only the books we choose !

Corpus example 3 : Twitter

- You can use the Twitter **RESTful API** to access data about both Twitter users and what they are tweeting about.
- To get started, you'll need to do the following things:
  - Set up a Twitter account if you don't have one already.
  - Using your Twitter account, you will need Apply to the Twitter Developer Account.
  - **Setup up a project** using this link. You would be asked to provide the project name, use case (similar to what you did while applying for the developer account), and a project description.
  - Once you've finished the preceding steps, you need to next **create an App**. It will be within the project you created in the previous step. *Important: the name of the app you're creating must not be duplicated or else you may receive an error.*
  - On the next screen, you'll be presented with keys & tokens i.e. API Key, API Key Secret, and Bearer Token. *Important: Please save these on your local machine, you will be using it later.*
  - Import the tweepy package.

```
import os
import tweepy as tw
import pandas as pd
```

See : https://www.earthdatascience.org/courses/use-data-open-source-python/intro-to-apis/twitter-data-in-python/

Corpus Data Management

- Usually, Text mining studies are applied to corpora containing thousands or tens of thousands of documents comprising gigabytes of data. We can also assume that it will require a preliminary steps of cleaning and pre-processing…
- The simplest and most common method of organizing and managing a text-based corpus is to store individual documents in a file system on disk. By maintaining each document as its own file, corpus readers can seek quickly to different subsets of documents and processing can be parallelized, with each process taking a different subset of documents.
- Data products often employ write-once, read-many (WORM) storage as an inter-mediate data management layer between ingestion and pre-processing as shown in Figure 2-2. WORM stores provide streaming read accesses to raw data in a repeatable and scalable fashion, and pre-processed data can be reanalysed without reingestion, allowing new hypotheses to be easily explored on the raw data format

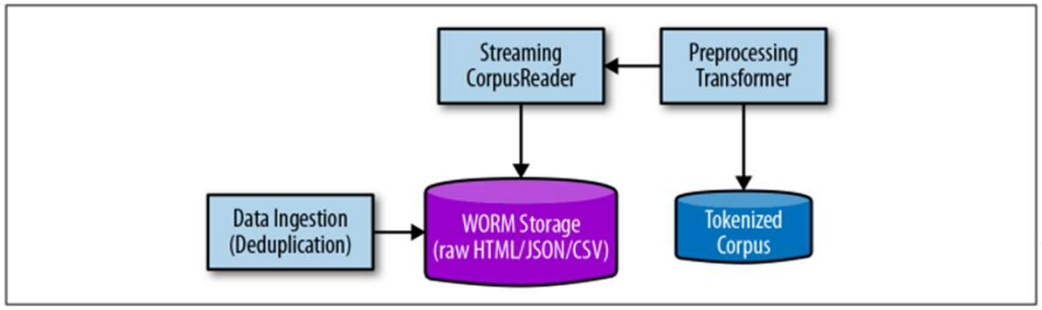


Figure 2-2. WORM storage supports an intermediate wrangling step

Lets do some Python !

For next week :

- Repository in GitHub : https://github.com/lserrar/LZML041
- Créer un compte personnel Github, et m'envoyer votre identifiant !
- M'envoyer un email sur loubna.serrar@gmail.com pour me confirmer l'installation de Python, et le compte Github avec une capture d'écran des résultats du notebook Homework1_Test : 2pts si avant le 17 février.
- Les slides du cours et les notebooks y seront disponible pour chaque séance