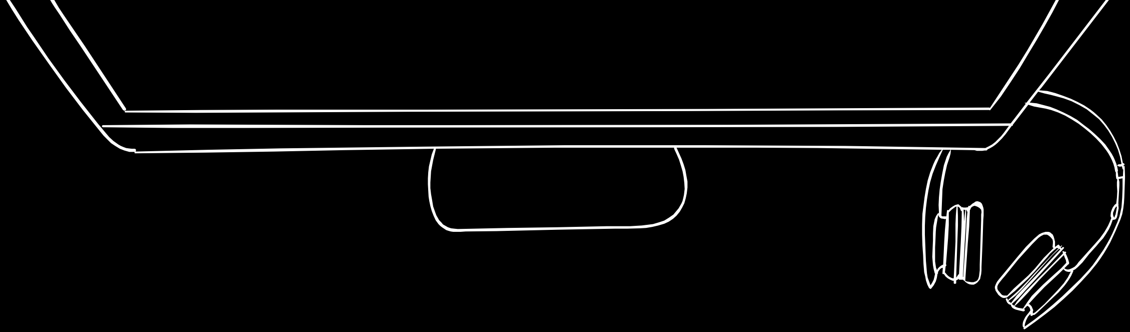
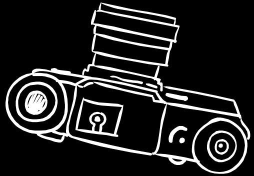
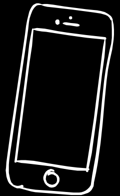
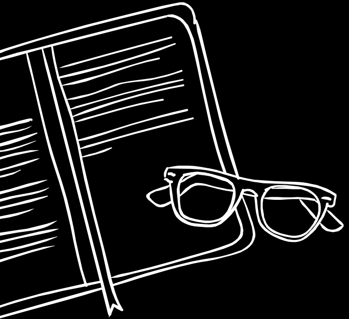
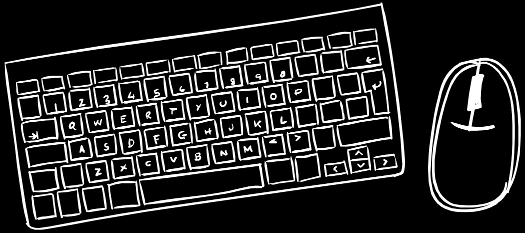
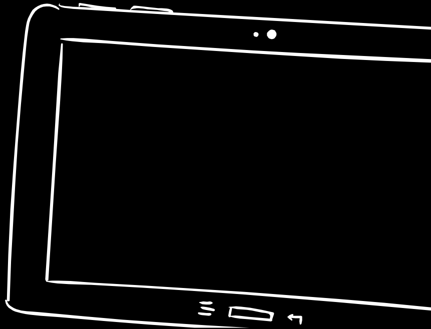



NLP AND TEXT MINING

By : Sarvesh Meenowa

Date : 09/12/2021

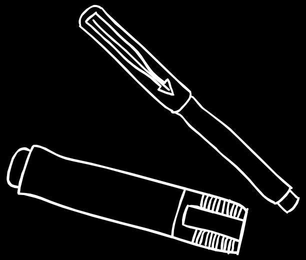
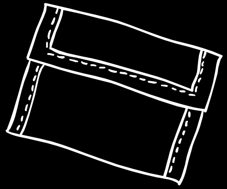
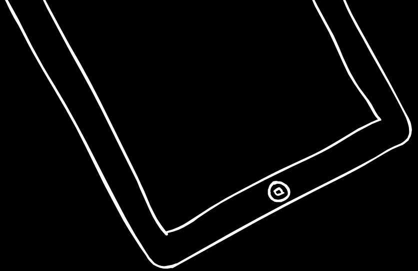
Introduction

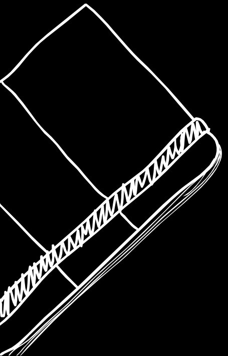
- ✖ Text mining has manifold applications. Companies leverage text analytics to update service offerings, improve compliance, anticipate public relations disasters, and so forth.
- ✖ The goals in this projects regarding text mining are to :
  - Detect languages using stopwords
  - Use the TF-IDF metric
  - Train and use a bigram and a 3-gram model
  - Use TF-IDF and cosine to assess similarity between the documents.
- ✖ Natural language processing has grown over the past decade, with products such as Siri, Alexa, and Google Voice Search using NLP to understand and respond to user requests.
- ✖ The goal of NLP in this project are to :
  - Design a strategy to automatically create the necessary data for MCQ exercises (target language is English)

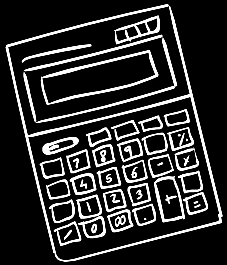
Outline

- Text mining
  - Language detection using stopwords
  - TF-IDF
  - N-GRAM
  - Similarity between documents with cosine similarity and TF-IDF
- NLP
  - POS-Tagging
  - Verb conjugation

Text mining - Language detection

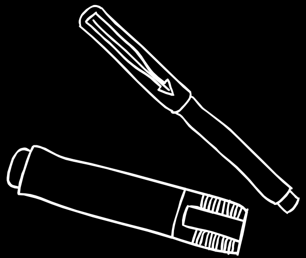
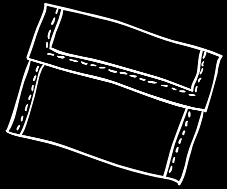
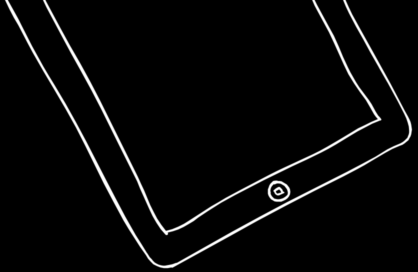






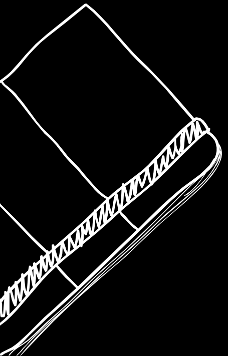


Language detection using stopwords

- Use NLTK (in python) corpus of stopwords(commonly used words in a language - which are of very little use)
- Tokenize using wordpunct_tokenize function and lowercase all splitted tokens
  - all punctuations and words separated into separate tokens
- Calculate probability of given text to be written in several languages and return the highest scored
  - uses a stopwords based approach
  - counting how many unique stopwords are seen.

Text mining - TF-IDF



What is Term frequency-inverse document frequency(TF-IDF) ?

- ✖ Proportional to term frequency
- ✖ Inverse function of the number of documents in which it occurs
- ✖ TF-IDF = term frequency * inverse document_frequency

$$w_{i,j} = tf_{i,j} \cdot \log\left(\frac{N}{df_i}\right)$$

$w_{i,j} \rightarrow$ weight of term i in document j

$tf_{i,j} \rightarrow term\ frequency\ of\ term\ i\ in\ document\ j$

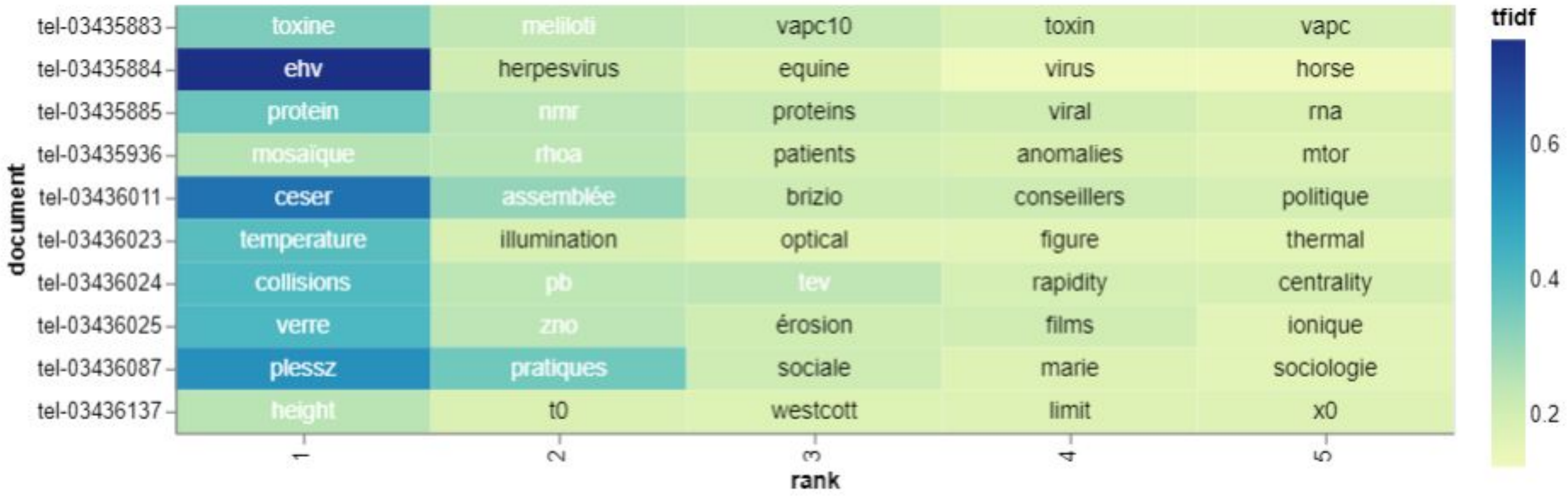
$N \rightarrow number\ of\ documents\ in\ the\ corpus$

$df_i \rightarrow number\ of\ documents\ cotaining\ term\ i$

How to implement TF-IDF in Python?

- ✖ Create a list of all the file paths of text files in relevant directory + list of all the titles
- ✖ Fit TfidfVectorizer (Transforms text to feature vectors that can be used as input to estimator) on the text files
- ✖ Make a DataFrame out of the resulting **TF-IDF** vector,
  - ○ Set “feature names” or words as columns
  - ○ Set the titles as rows
- ✖ Add column for document frequency i.e number of times word appears in all documents
- ✖ Reorganize the DataFrame ⟶ words are in rows rather than columns.
- ✖ Find the top 5 words with the highest **TF-IDF** for every document

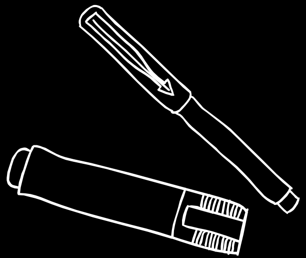
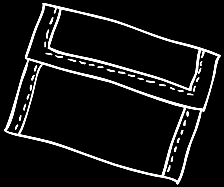
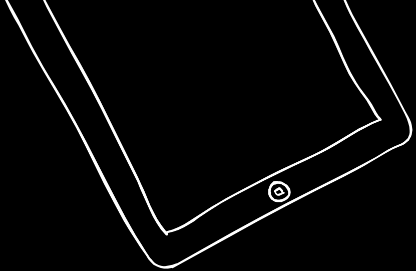
TF-IDF on 10 documents

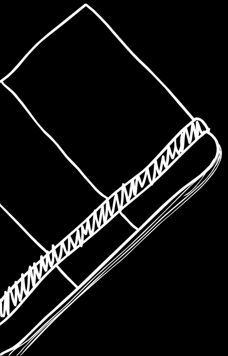


document	1	2	3	4	5
tel-03435883	toxine	meliloti	vapc10	toxin	vapc
tel-03435884	ehv	herpesvirus	equine	virus	horse
tel-03435885	protein	nmr	proteins	viral	rna
tel-03435936	mosaïque	rhoa	patients	anomalies	mtor
tel-03436011	ceser	assemblée	brizio	conseillers	politique
tel-03436023	temperature	illumination	optical	figure	thermal
tel-03436024	collisions	pb	tev	rapidity	centrality
tel-03436025	verre	zno	érosion	films	ionique
tel-03436087	plessz	pratiques	sociale	marie	sociologie
tel-03436137	height	t0	westcott	limit	x0

Figure 1 : Heatmap of the top 5 words with highest TF-IDF from 10 documents

Text mining - N-gram



N-GRAM

- ✖ Contiguous sequence of n elements (or words) in a given document.
- ✖ n = 1 ⟶ bag-of-words : [“for you a thousand times over”]
- ✖ n = 2, n-grams: ['for you','you a','a thousand','thousand times', 'times over']
- ✖ n = 3 ,n-grams: ['for you a','you a thousand','a thousand times','thousand times over']

Applications

- Sentence completion
- Spelling correction
- Machine translation correction

Building n-gram models using scikit-learn

- only bigrams :
  bigrams = tfidfVectorizer(ngram_range=(2,2))
- unigrams, bigrams and trigrams :
  ngrams = tfidfVectorizer(ngram_range=(1,3))

Shortcomings

- Curse of dimensionality
- Higher order n-grams are rare
- Keep n small

BI-GRAM

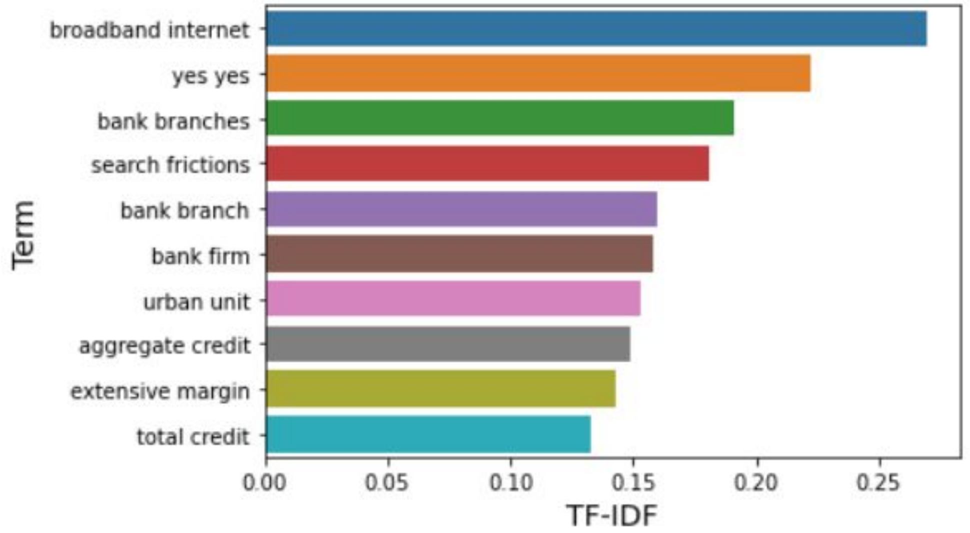


Figure 2 : Bar plot of bigrams of a document with top 10 highest TF-IDF

3-Gram

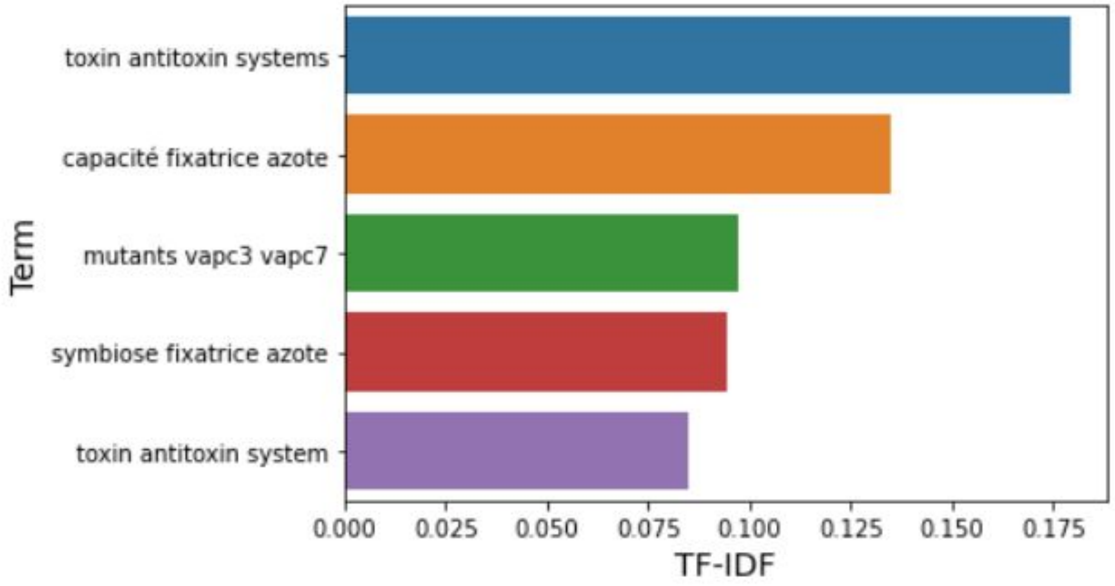
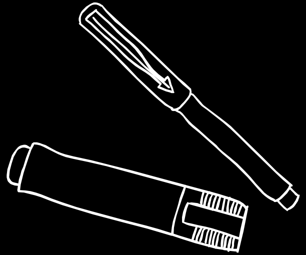
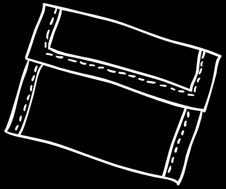
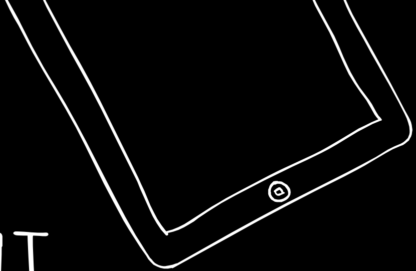
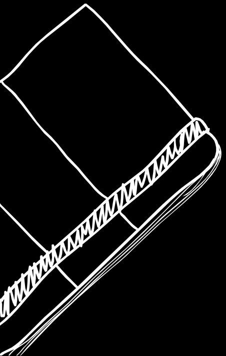


Figure 3 : Bar plot of 3-grams of a document with top 5 highest TF-IDF

Text mining - document similarity



Similarity between documents

Cosine Similarity

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|}$$

→ Count the word occurrence in each document using tf-idf vectorizer

→ Generated vector matrix (sparse matrix)

→ Apply cosine similarity on vector matrix

Jaccard Similarity

$$J(doc_1, doc_2) = \frac{doc_1 \cap doc_2}{doc_1 \cup doc_2}$$

→ List the unique words in a document

→ Find the intersection of words list of the docs

→ Find the union of words list of the docs

→ Calculate Jaccard similarity score

✖ using length of intersection set divided by length of union set

Force directed network

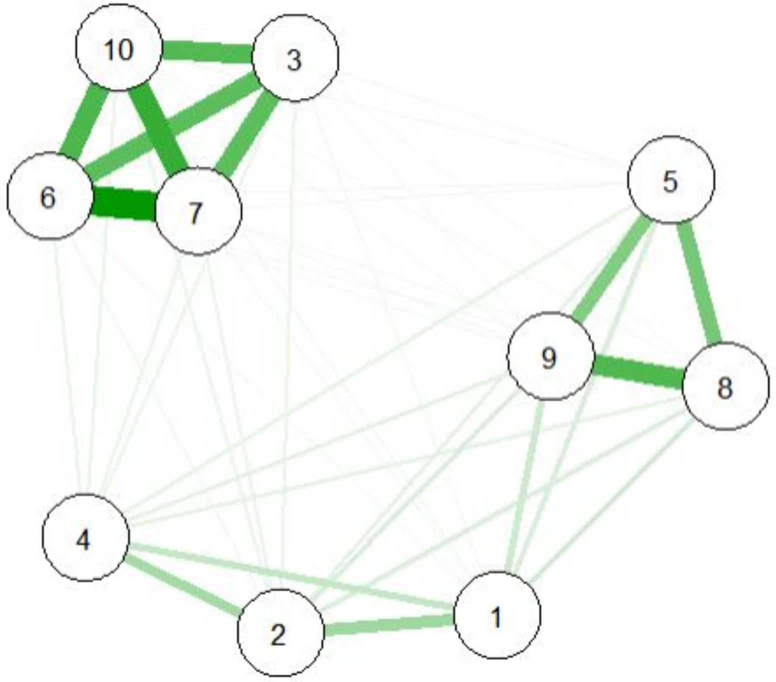


Figure 4 : Force directed graph of the cosine similarity matrix of 10 theses

[1]"Les systèmes Toxine-Antitoxine VapBC : des régulateurs de la symbiose fixatrice d'azote Rhizobium-Légumineuse,The VapBC Toxin-Antitoxin systems : regulators of the nitrogen-fixing symbiosis Rhizobium-Legume"

[2] "Étude phylogénique de souches d'alphaherpèsvirus isolées chez les équidés français et développement d'un outil innovant pour la mesure des anticorps neutralisants après infection ou vaccination,Phylogenic study of equid alphaherpesvirus strains isolated in France and development of an innovative assay for the measurement of neutralising antibodies after infection or vaccination"

[3] "Multi-scale studies of Measeles virus nucleocapsid assembly,Etudes multi-échelles de l'assemblage de la nucléocapside du virus de la rougeole"

[4] "Caractérisation génomique des anomalies de la pigmentation cutanée en mosaïque,Genomic characterization of mosaic cutaneous pigmentary disorders"

[5 "Le Conseil économique\\, social et environnemental régional : assemblée du dialogue des intérêts organisés dans la région,The Conseil économique\\, social et environnemental régional : an assembly of dialogue between organised interests in the Région"

[6] "Optical Developments for Microscale Measurement and Control of Temperature in Optogenetics,Développements optiques pour la mesure et le contrôle micrométrique de la température en optogénétique"

[7] "Z-boson and double charm production with ALICE at the LHC,Production des bosons Z et du double charme avec ALICE auprès du LHC"

[8] "Caractérisation opto-mécanique du verre traité par des méthodes thermo-chimiques,Opto-mechanical characterization of glass treated by thermochemical methods"

[9] "La Dynamique sociale des pratiques : stratification sociale\\, changement social et consommation alimentaire,The Social Dynamics of Practices : social stratification\\, social change and food consumption"

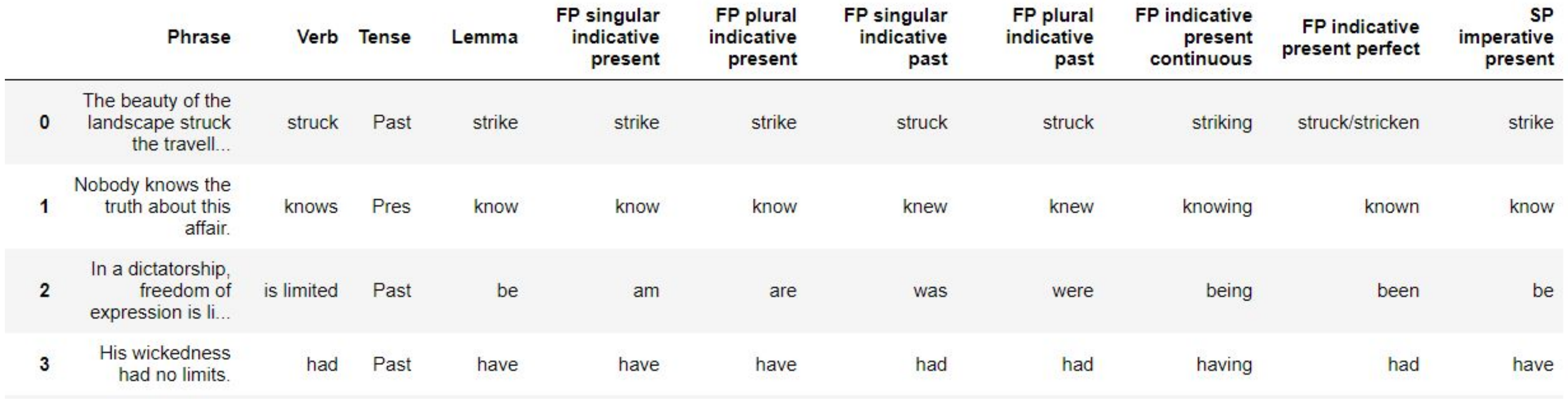
[10] "Random surface growth models : hydrodynamic limits and fluctuations,Modèles de croissance de surfaces aléatoires : limites hydrodynamiques et fluctuations"

Natural language processing

POS-TAGGING

- ✖ Process of marking up a word in a corpus to a corresponding part of a speech tag, based on its context and definition
- ✖ Strategy to automatically create the necessary data for MCQ exercises in English :
  - ○ Use stanza, spacy-stanza, Spacy
  - ○ Download and load model corresponding to target language
  - ○ Use token.pos to detect verbs and nouns
  - ○ Use token.morph to get the tense of the verbs and grammatical number
  - ○ Use token.lemma to get the lemmatized form of the verbs
  - ○ Use mlconjug3 to conjugate the lemmatized verbs into wanted tense

Final database format



	Phrase	Verb	Tense	Lemma	FP singular indicative present	FP plural indicative present	FP singular indicative past	FP plural indicative past	FP indicative present continuous	FP indicative present perfect	SP imperative present
0	The beauty of the landscape struck the travell...	struck	Past	strike	strike	strike	struck	struck	striking	struck/stricken	strike
1	Nobody knows the truth about this affair.	knows	Pres	know	know	know	knew	knew	knowing	known	know
2	In a dictatorship, freedom of expression is li...	is limited	Past	be	am	are	was	were	being	been	be
3	His wickedness had no limits.	had	Past	have	have	have	had	had	having	had	have

Figure 4 : Extract of the final form of the database to generate exercises regarding verbs

REFERENCES

Cosine Similarity—Text Similarity Metric. (2019, September 29). *Machine Learning Tutorials*.

https://studymachinelearning.com/cosine-similarity-text-similarity-metric/

Courses—DataCamp Learn. (n.d.).,

https://app.datacamp.com/learn/courses/feature-engineering-for-nlp-in-python

Detecting text language with python and NLTK - Alejandro Nolla—Z0mbiehunt3r. (n.d.).

https://blog.alejandronolla.com/2013/05/15/detecting-text-language-with-python-and-nltk/

TF-IDF with Scikit-Learn—Introduction to Cultural Analytics & Python. (n.d.).

https://melaniewalsh.github.io/Intro-Cultural-Analytics/05-Text-Analysis/03-TF-IDF-Scikit-Learn.html