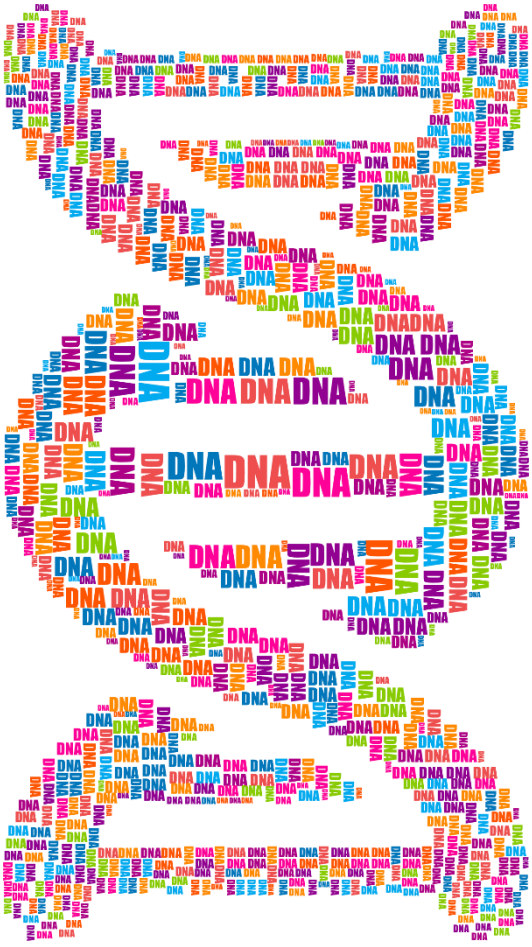
27/06/2018

User guide



UCSI software project

University of Colorado Summer Internship (UCSI) software project user guide

Table des matières

Introduction

This app is a visualization tool for Natural Language Processing. It has two purposes:

- Process textual data and analyze it
- Visualize the data obtained after analysis

The app should help researchers in linguistics understanding the structure of the documents they study. It provides a set of statistical metrics, sentence structure analysis and other analysis. It then represents them in order to extract useful information from it. It is built on Shneiderman's mantra “Overview first, zoom and filter, then details-on-demand”. It should cover a wide range of the possible analysis in Natural Language Processing and present them efficiently. The app is composed of different screens that will enable you to do your analysis. Firstly, you will be able to download your own data into the app, and then you will be able to perform your analysis, with a pre-processing part and then a statistical analysis.

I. Installation

a. Prerequisites

In order to use the application, you need to have:

- R version: R version 3.5.0 (2018-04-23)
- Java
- RStudio

b. Installation

To install the app, you just need to run install_packages.R file and it will download

the specific packages needed to use the app. This file is also called in the main.R file, when you use the application. However, it is better the first time to call the install_packages.R file to see what problems there are concerning the installation.

There are several bugs that you may encounter in this process. They are all described in …….. .

You might also need to install Rtools, which can be done at https://cran.r-project.org/bin/windows/Rtools/ . This can be of help if the installation fails and the error is not in the bugs reported.

II. Usage

a. Launch the application

You need to:

- Open the main.R file
- Choose the path of the GitHub repository on your computer. You need to change the antislashes (\ ) in slashes (/) in the path you put. This path should be the path before the folder you just downloaded on GitHub. For instance, if the path to the folder "app" in the folder you downloaded on GitHub is "C:/Users/Projet/Internship_NLP_CU/app", then you should put "C:/Users/Projet/Internship_NLP_CU" as my_path.
- Launch the file in RStudio. In order to do so, select all the code in the file, by clicking four time on the code or doing `ctrl` + `A` , and then do the command `ctrl` + `enter` to launch it.
- To close the application, close the application window and then press `esc` on RStudio.

b. Using the app

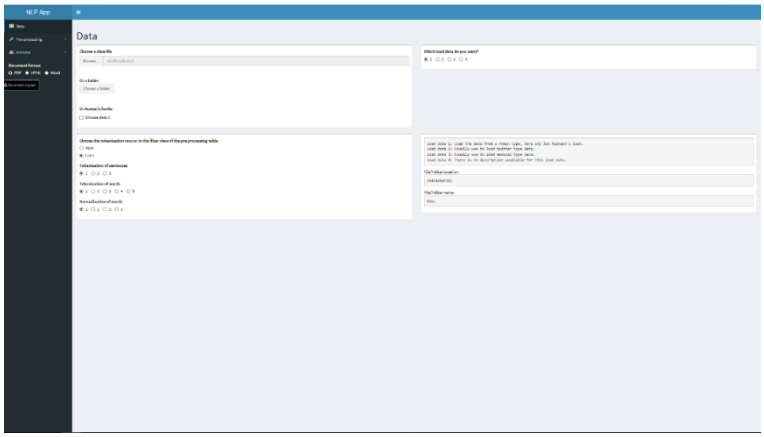
The app consists of multiple screens that you can select. Each of this screen has a function in the analysis. Firstly, to use the app, you can either upload your data in the app in the first screen that appears or use a demo data, that are an extract from Jane Austens books. Then, by passing from screen to screen with a tab panel on the left of the screen, you will be able to perform your analysis. The screens are divided in sections, Pre-Processing and Analysis. The first consists of helping you to choose part of the data you've uploaded and the tokenization for the sentence, word and normalization. The second section consists of the analysis, you can select sentences to analyze, then a word in this sentences. This lead to a final screen that is the end of the process. When you finish your analysis, you can download a report with all the information of your analysis (see example of report here .....;bazjszdiupceiupdceiuedcciu).

1. Data

Files used for this screen:

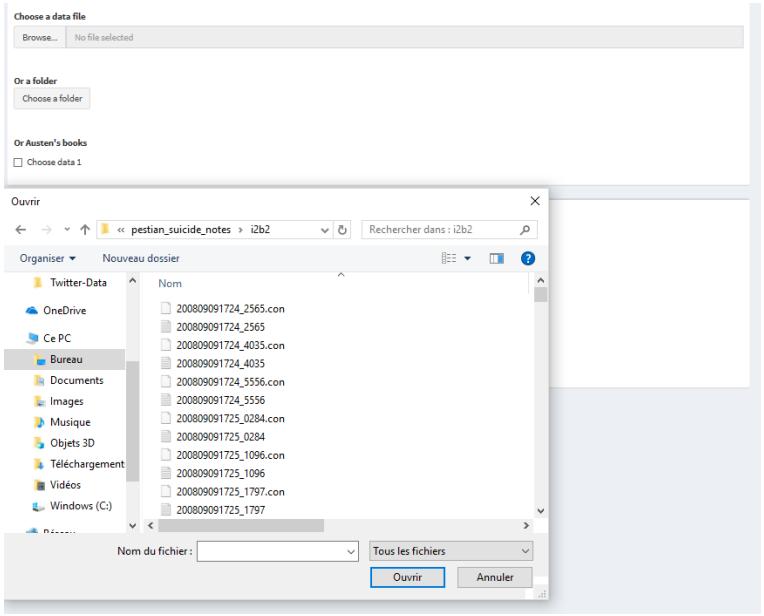
- Intership_NLP_CU-master\description\ description_type_data.R
- Intership_NLP_CU-master\load_data
- Intership_NLP_CU-master\app
- Particularly: line 5 to 106 from Intership_NLP_CU-master\app\server.R

The data screen is mainly used to load your data into the app. You can choose how to read it, and if you want, which tokenization to apply. If you want to use the app without putting your data in it, choose “Choose data 1” and “1” in “Which load data do you want?”.



This first box is used to load your data. Depending on how to read it, you can either choose a folder or a file. If you want just to test the app, you can also choose an extract of Jane Austen’s books.

The folder option can be used to read different files that are in the same folder and merge them. For example, you can analyze many last words of people which are all in one file.



One of the goal of the app is to help you choose the right tokenization and see the impact of this choice. However, you can also only use the app to analyze the text you want. So if you already know which tokenization you want to use, you can choose it in this screen by checking “Now”. It will automatically process the text with the right tokenization in the analysis. In the default case, you choose the tokenization in the filter part of the pre-processing (“Later”). Each number of tokenization corresponds to a file number located in Intership_NLP_CU-master\preprocessing folder.

This is the box where you can choose which function will read your data. There might not be an appropriate function here for your data. In this case, read ……. to learn how to add one.

You can see below the checkboxes the description of each function, along with the input needed. You can also see the file location and name of the file you selected or the folder location if you have chosen this option.

2. Pre-processing

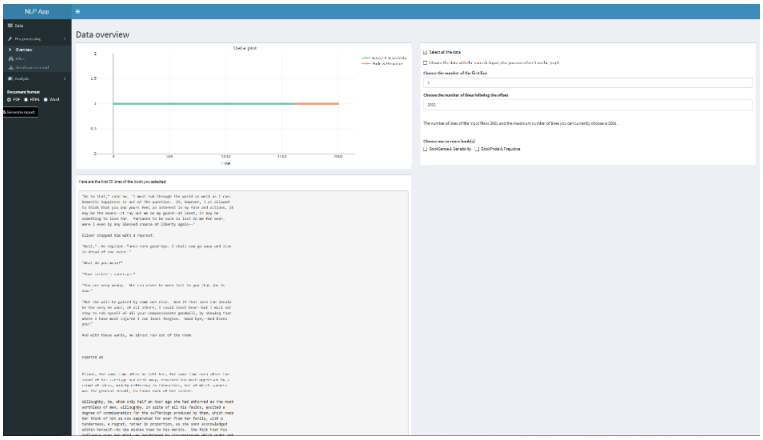
This part is to pre-process the text. It will enable you to choose more precisely which part of the text you want to study, show the importance of the choice of tokenizations and see how regular the data is. At the end of this process, you will have chosen the tokenizations you want.

a. Overview

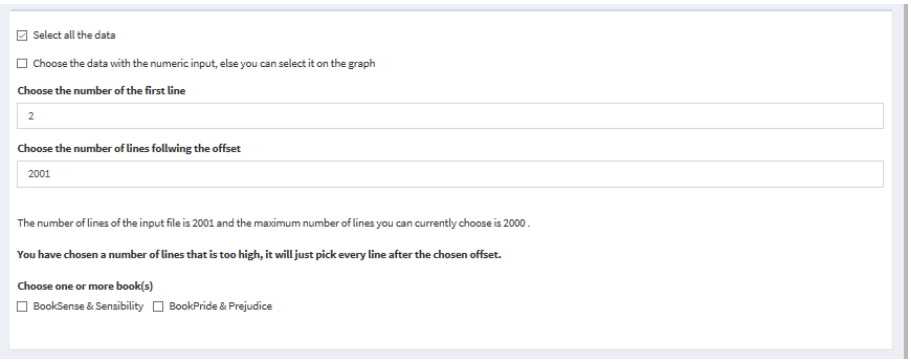
Files used for this screen:

- Intership_NLP_CU-master\app
- Particularly: line 106 to 270 from Intership_NLP_CU-master\app\server.R

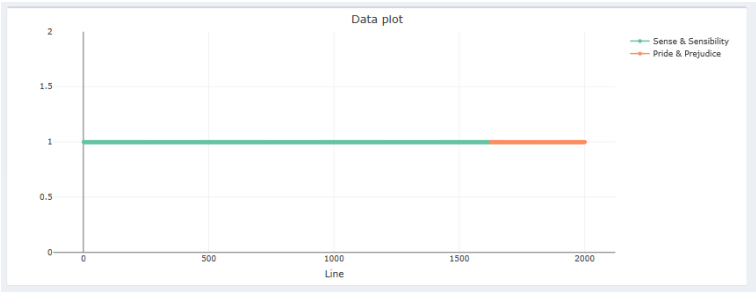
The overview screen is the one to use to choose precisely which part of the text to analyze. You have several ways to select it that will be explained just below. Different parts of the text can have different linguistics particularities. And this overview should enable you to highlight that.



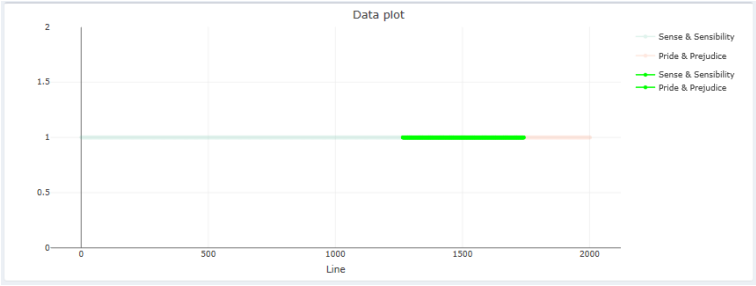
In this box, you have three options to choose the data. There can't be two options at the same time. If you check "Select all the data", the text will be analyzed totally. This is the default option. You can also select which lines you want. To do that, check the second check box, and then put the number of the first line and the offset you want. There is a message telling you how many lines there are in the text and the maximum number of lines you can choose. If you have chosen a number of lines too high compared to the number of lines and the offset, a bold message will appear warning you of this problem. The third way to select your data is to select it by book. You can select as many books as you like with the checkboxes. If you didn't put any book in your input data, there will just be a default name of book for the whole data ("BookAllTheSame").



The fourth and last way to select data is to select it manually. You can use the mouse to select it on the plot directly. If there is a checkbox selected, you must select once to decheck the checkbox and then another time to select the data for real. The advantage of this method is that you can see which book to select and which part of that book you select (the beginning, the end…). The different colors on the plot indicate different books.

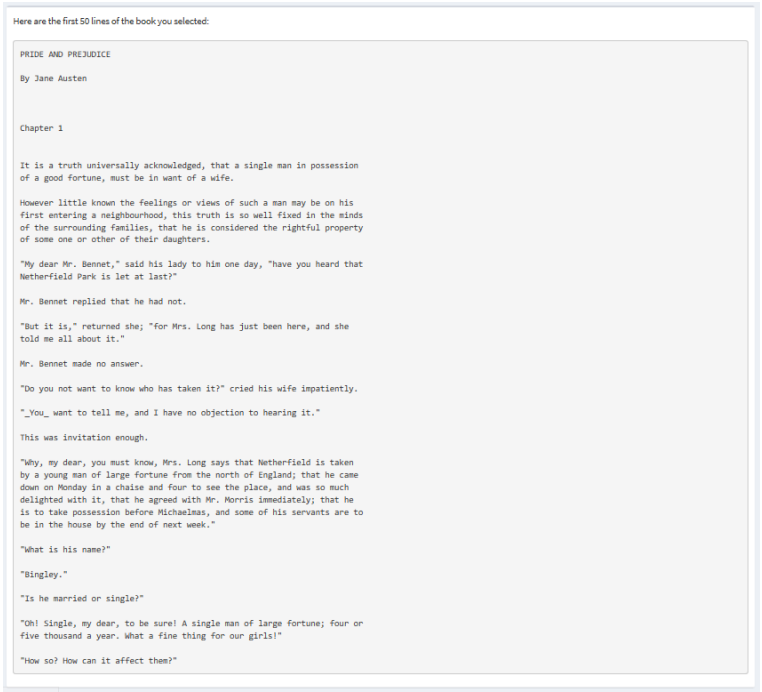


The number of lines corresponds to the size of the text. And the abscissa of the plot is the number of lines.



If you can't manage to select on the plot, see ..... . It is probably due to the browser you're using.

You can see just on the left what happens when you select directly on the plot.



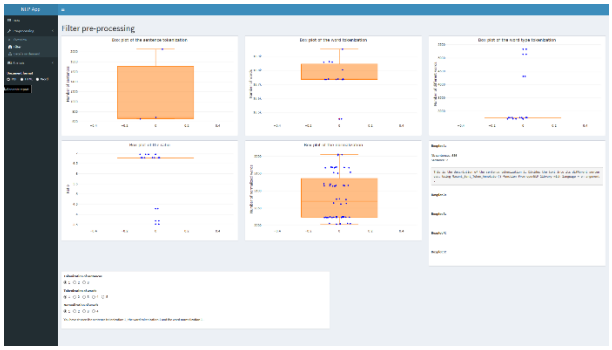
The last feature of the overview is the box on the left here. It prints the first 50 lines of the selected text. It is a good way to see where you're in the text. It creates a link between the number, the books and the real text. Usually, when analyzing a text, there is a shifting towards the numbers (number of words, frequency, line number…). This feature enables you to not forget the text you're analyzing. By visualizing the text, it enables you to be more precise in your selection. For example, if you want to select just the introduction, although you don't know exactly the line it ends, you can see where it is.

b. Filter

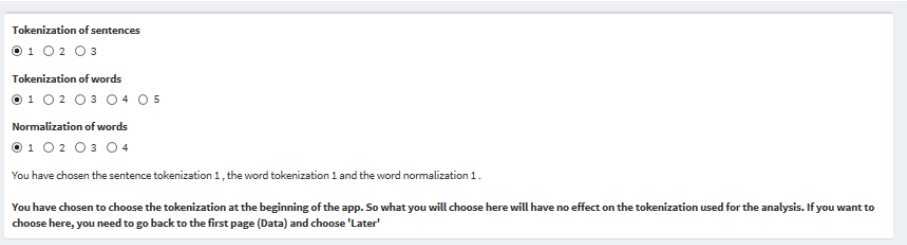
Files used for this screen:

- Internship_NLP_CU-master\backend_analysis\token_boxplot
- Internship_NLP_CU-master\backend_analysis\modulo_not_null
- Internship_NLP_CU-master\preprocessing
- Intership_NLP_CU-master\app
- Particularly: line 270 to 501 from Intership_NLP_CU-master\app\server.R

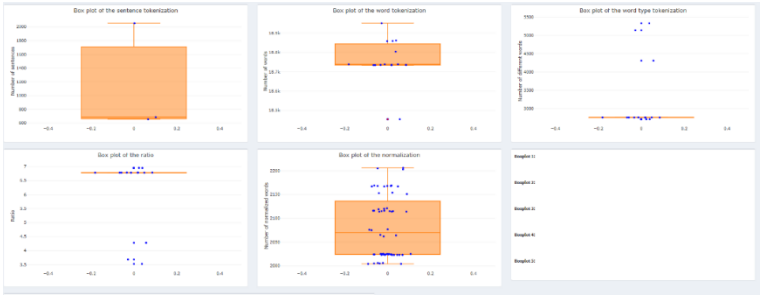
The filter screen is the one to see the impact of the choice of tokenization. This is also where you can make your choices of tokenzations for the rest of the analysis considering the results of the boxplots. It will enable you to choose wisely your tokenization, which is usually a big problem to face.



This is the box to choose the tokenization. The choices are automatically taken from the files where the tokenzations are (see Internship_NLP_CU-master\preprocessing for more details). If you have chosen “Now” on the data screen, a message will appear saying that the choices you make here will have no impact on the rest of the analysis unless you change your choice on the data screen. It will also always show you your choices of tokenization.



The first boxplot shows the variation of the number of sentences depending on the tokenization_sentence function used.



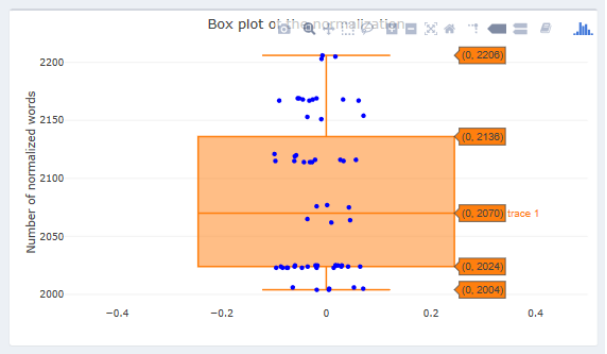
This boxplot shows the variation of the number of word occurrences depending on the tokenization_word function used. That means the total number of words in the text selected (with repetition).

This boxplot represents the variation of the number of word type depending on the tokenization_word function used. That means the number of distinct words in the text select (without repetition).

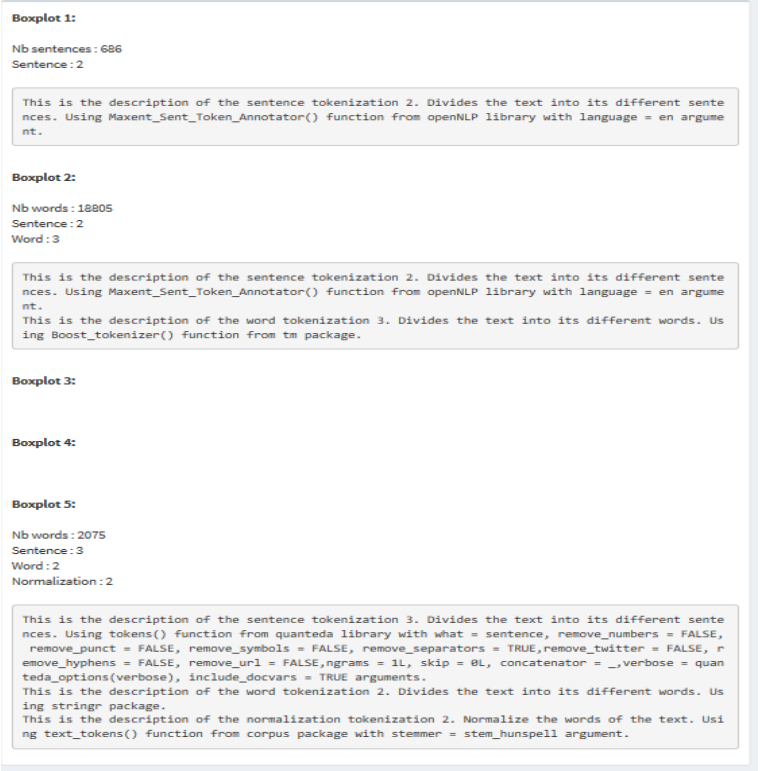
This boxplot shows the variation of ratio (which is the word occurrences divided by the number of word types) depending on the tokenization_word used. The ratio is a very meaningful information about the text. A huge variation of this data regarding on the tokenization _word could be a real problem, in a sense that a really accurate value for the ratio has a really specific meaning.

This boxplot shows the variation of the number of type words after normalization depending on the normalization function used. Normalization is a process during which each word is transform into a single canonical form those forms are regrouped so they appear just ones (e.g. ‘go’, ‘goes’, ‘going’, ‘gone’ and ‘went’ will be associated to ‘go’).

This are the five boxplots. On all of them, you have two interesting features. The first one is when you hover over the boxplots you can see the lower whisker, first quartile, median, third quartile and the upper whisker. Each point corresponds to a number of a certain tokenized identity after applying a certain tokenization (e.g. the number of words after applying tokernization_word_1). The abscissa of a point is determined randomly in order to be distinguishable from the others.



The second interesting feature is that if you click on one the blue points, you will have information about these one appearing on the left of the screen. You will have the ordinate of the point selected, its tokenization(s) and the description of its tokenization(s). The description should enable you to better understand which point and whether it is pertinent or not.

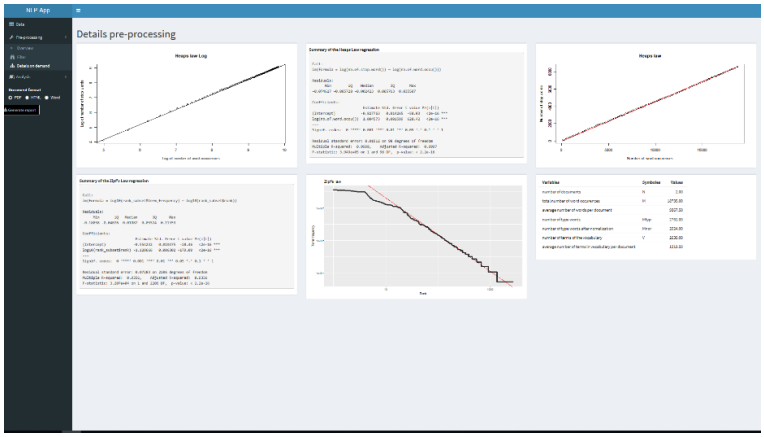


c. Details on demand

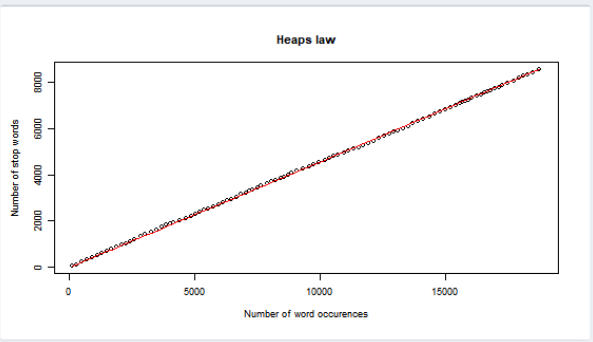
Files used for this screen:

- Internship_NLP_CU-master\backend_analysis\heaps_law.R
- Internship_NLP_CU-master\backend_analysis\zipfs_law.R
- Internship_NLP_CU-master\ backend_analysis\table_info.R
- Internship_NLP_CU-master\ backend_analysis\after_choose_token.R
- Intership_NLP_CU-master\app
- Particularly: line 501 to 604 from Intership_NLP_CU-master\app\server.R

The details on demand screen is used to see how regular the data is. It tests that by plotting two important laws that verify this. It also shows summary tables with interesting numbers from the tokenization. The data analyzed here is the data with the tokenizations chosen before.



In the screen, there is three plots. The first two ones are about Heaps law. One is in log and shows how well the regression is. The other one is Heaps law itself. Heaps law is a law that verify that the bigger the number of occurrences there are the bigger the vocabulary is.



Along with this plots, there is a summary if the linear regression. It enables you to see how well the regression is and if Heaps law works well. And know the value of the empirical formula of Heaps law. If you want to know more about summaries in R, go to [this link](#).

Summary of the Heaps Law regression

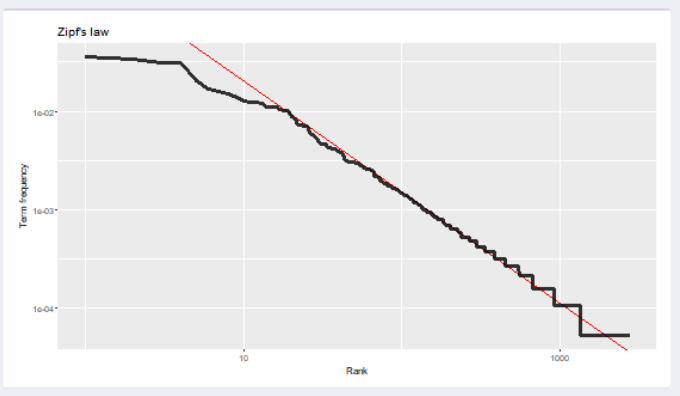
```
Call:
lm(formula = log(nb.of.stop.word()) ~ log(nb.of.word.occu()))

Residuals:
      Min        1Q    Median        3Q       Max
-0.074617 -0.005728 -0.002423  0.005763  0.035507

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)          -0.827763   0.014265  -58.03   <2e-16 ***
log(nb.of.word.occu())  1.004573   0.001599  628.42   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.01512 on 98 degrees of freedom
Multiple R-squared:  0.9998,    Adjusted R-squared:  0.9997
F-statistic: 3.949e+05 on 1 and 98 DF,  p-value: < 2.2e-16
```

The last plot is Zipf's law. This law indicates that the frequency of occurrence for a word is inversely proportional to its rank. The rank is obtained when we sort the frequency by decreasing order. There is also a summary next to it to see the regression. It is the same idea as for Heaps law.



Finally, this table shows interesting numbers concerning the chosen tokenization. It summarizes and go further in the analysis. If you want to add other numbers, go see ..... to learn how to do that.

Variables	Symboles	Values
number of documents	N	2.00
total number of word occurences	M	18735.00
average number of words per document		9367.50
number of type words	Mtyp	2761.00
number of type words after nomalization	Mnor	2024.00
number of terms of the vocabulary	V	2639.00
average number of terms in vocabulary per document		1319.50

3. Analysis

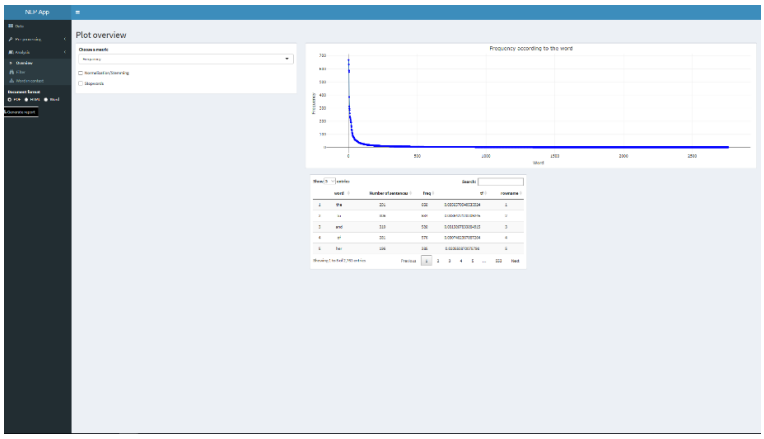
The goal of this part is to analyze a text. It should link two views of the language that are the symbolic one and the statistical one. The overview is a statistical analysis, and the "word in context" is a symbolic analysis while the filter is the link between the two. Here the statistical analysis is a macroscopic analysis while the symbolic one is a microscopic one. That is one of the reason Schneiderman's mantra was chosen to implement this app.

a. Overview

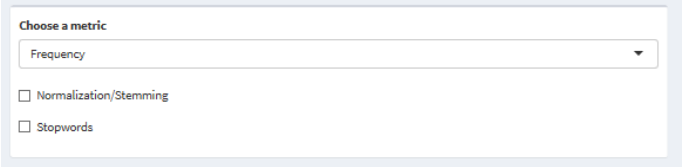
Files used for this screen:

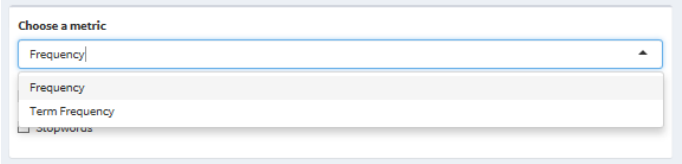
- Internship_NLP_CU-master\preprocessing\stop_word
- Internship_NLP_CU-master\preprocessing\normalization
- Intership_NLP_CU-master\app
- Particularly: line 605 to 653 from Intership_NLP_CU-master\app\server.R

This screen enables you to do several tasks. You can choose a metric, and then according to this one, you can select the words you want to focus on for the next step. In order to do this, there is also a table in which you can search more precisely.

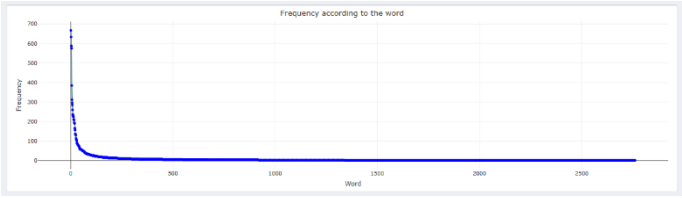


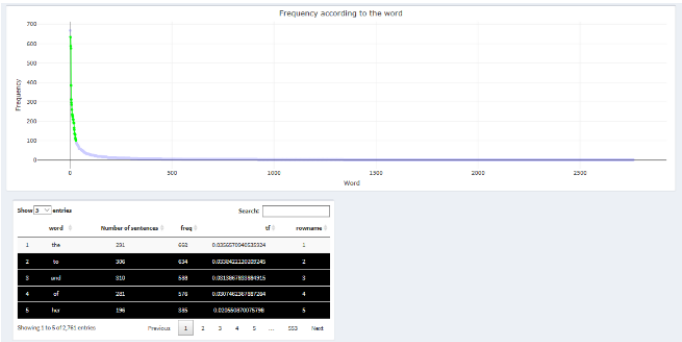
On this box, you can choose a metric (frequency or term frequency) on the select input button. To add another metric, see ……. You can also select if you want your text to be normalized, without stopwords or both, by checking the box below the select input button.



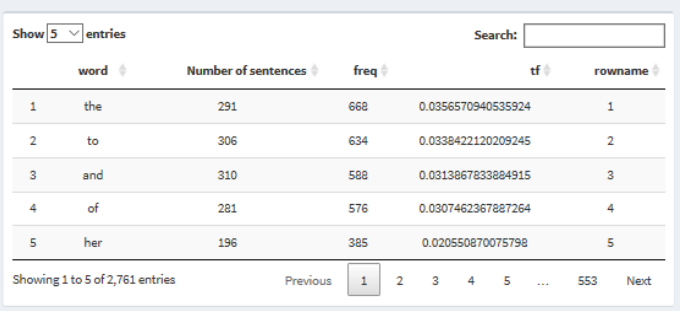


This box shows an interactive plot, that changes accordingly to the chosen metric. In the abscissa, the words are sorted by frequency. In almost every text, the frequency curve should look like the one on the right figure. You should select data points on the plot. It is a necessary step to continue the analysis afterward. Indeed, the selected words will be the one to be analyzed. In order to deselect them, you need to double-click on the plot. They will automatically appear on black in the data table, while the non-selected points will appear on white in the data table. You can see this on the right figure.





Finally, this box is a data table in which you can search very precisely. You can sort by every column (by frequency, tf…). You can search for a particular word or value. You can also choose to show more entries. It is a powerful tool to complete your statistical analysis.


Show 5 entries Search:

	word	Number of sentences	freq	tf	rowname
1	the	291	668	0.0356570940535924	1
2	to	306	634	0.0338422120209245	2
3	and	310	588	0.0313867833884915	3
4	of	281	576	0.0307462367887264	4
5	her	196	385	0.020550870075798	5

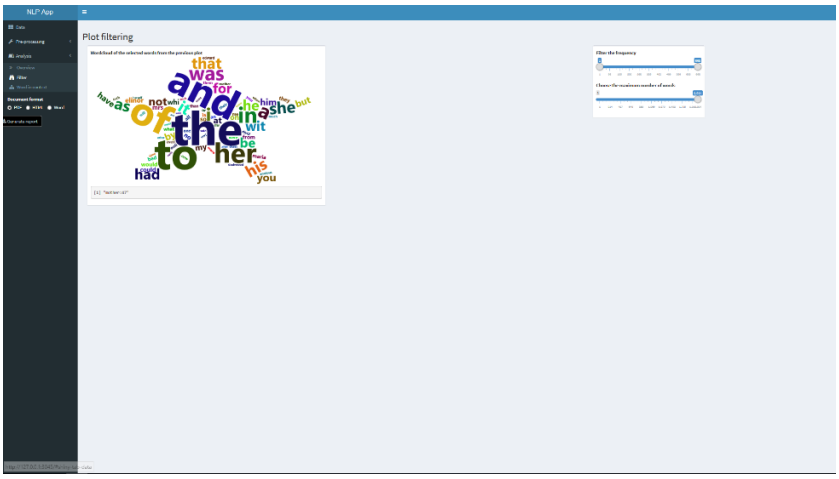
Showing 1 to 5 of 2,761 entries Previous 1 2 3 4 5 … 553 Next

b. *Filter*

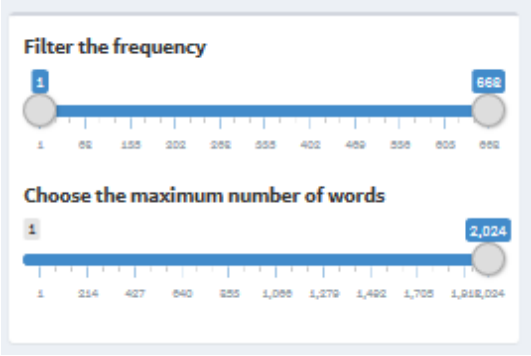
Files used for this screen:

- Intership_NLP_CU-master\app
- Particularly: line 654 to 694 from Intership_NLP_CU-master\app\server.R

The filter screen enables you to choose more precisely which word you want to study. It also helps with a wordcloud of the words. This is both a filter to analyze in more depth and a tool to deduce interesting points.



This box enables you to choose a frequency range for the words in the cloud and a maximum number of words in the cloud with the slider inputs. This is a good tool to filter even more your text. There is one common bug that appears when you select only one word. It is described in …… .



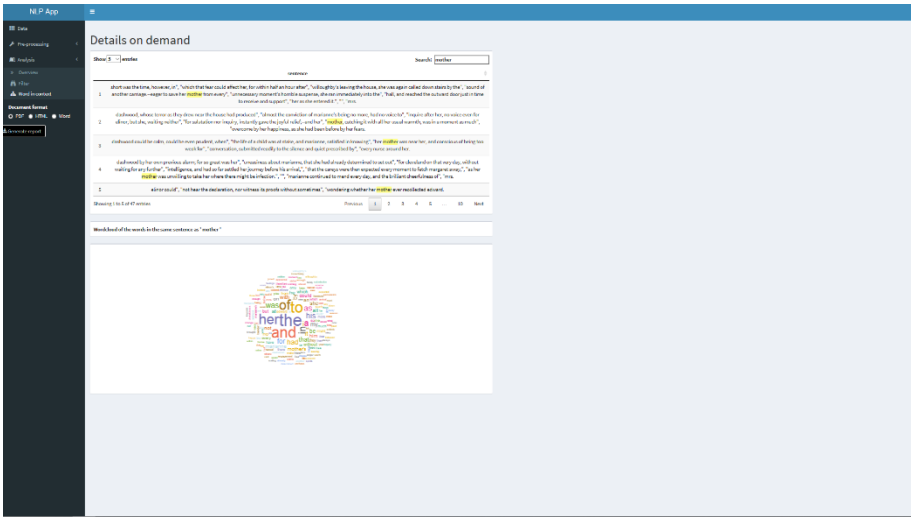
This box is a wordcloud. The sizes of the words are proportional to the frequency. The words in this wordcloud are the ones selected in the previous plot (in the “Overview”). You hover on the words and see the associated frequencies. You can also click on a word and see it appears at the bottom of the word cloud with its associated frequency. You should click on a word you want to analyze. Indeed, the last step uses the selected word here to analyze more in depth.

c. Word in context

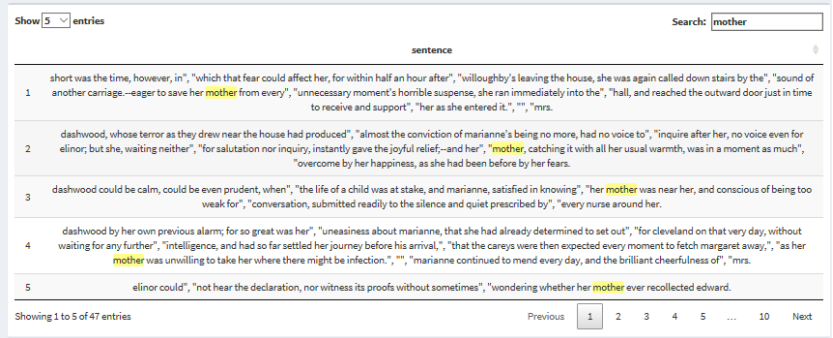
Files used for this screen:

- Internship_NLP_CU-master\backend_analysis\ wordcloud_data_func.R
- Intership_NLP_CU-master\app
- Particularly: line 695 to 762 from Intership_NLP_CU-master\app\server.R

This screen allows a more detail analysis link to a specific word. This word is the one chosen at the last step (“Filter”). You can see all the sentences it appears in, and all the words which are in the same sentence with their frequencies. This tool is the last part of the analysis and constitutes the symbolic analysis.



This box shows a table in which all the sentences which the chosen word in it appear. This selected word is highlighted in yellow. This enables a grammatical analysis and links the whole analysis to the text itself.

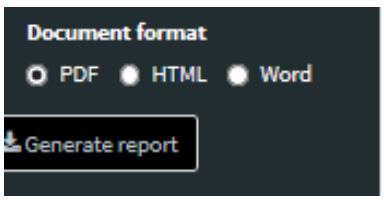


This box shows a wordcloud with all the words in the same sentence as the selected word (here 'mother'). The sizes of the words depend on their frequencies. However, these frequencies are not the same as earlier. They are the frequencies of this words in the sentences where the selected word appears.

4. Report

Files used for the report:

- Intership_NLP_CU-master\app
- Particularly: line 763 to 812 from Intership_NLP_CU-master\app\server.R
- And Intership_NLP_CU-master\app\report.Rmd



The app enables you to download a report of what you did during the session. This report includes the plots parameters that you chose during this session, the boxplots, the laws, the plots of the analysis overview, the plots, table and wordcloud of the selected points, and the sentences and wordcloud of the selected word. It is available in three output formats (PDF, HTML and WORD). It also contains a table of content. You can see an example of report in Internship_NLP_CU-master\guide. The power of this report is to have all the results of your analysis stored in your computer so that you can use them for your articles. It is also a good way to compare different documents. It is particularly easy with the reminder of the parameters you chose to reproduce it to another document. There is a common bug with the PDF that can occur that is described in ..... .

III. Add new features

a. Load data

To add a way to read a data, go to <your_path>\Internship_NLP_CU\load_data. Add a file with load_data_i.R for name, with i the next number which has not already been used (needs to have no gap beetween the numbers). For instance if there already are load_data_1.R, load_data_2.R and load_data_3.R in the folder, you will name your file load_data_4.R. Use the model below and fill it thanks to the information below.

```
#' @description ...    ①
#'
#' @param path A character ...    ②
#' @return original_book A tibble with two colums.
#' original_book$text is the lines of the text.
#' original_book$book is the part (e.g. chapter, different book...)
#' of the full text to which the lines belong
#'
#' @import ...    ③
#' @examples
#' ## library(...)    ④
#' ## original_books <- load.data.i(ex.path)

load.data.i ⑤ <- function(path) {

... ⑥

  return(original_book)
}
```

① Describe the particularity of this function to load data. For example, for which data it is used, which type of input need to be given. It has to be just on one line because the description is automatically collected to be displayed in the app.

② Describe the input. There are three possibilities: the path to a file, the path to a folder where all documents are read and merge, or a useless path not needed if the text is read in the app itself (e.g. Jane Austeen book).

③ and ④ Write the name of every package needed in your function.

⑤ Replace i by the number you used for the name of your file (load_data_i.R).

⑥ Write the code of your function. It has to read the text and return, a tibble with two columns (original_book):

- original_book$text is the lines of the text.

- original_book$book is the part (e.g. chapter, different book...) of the full text to which the lines belong

b. Tokenizer sentence

To add a way to tokenize sentences, go to <your_path>\Internship_NLP_CU\preprocessing\tokenizer_sentence. Add a file with tokenizer_sentence_i.R for name, with i the next number which has not already been used (needs to have no gap beetween the numbers). For instance if there already are tokenizer_sentence_1.R, tokenizer_sentence_2.R and tokenizer_sentence_3.R in the folder, you will name your file tokenizer_sentence_4.R. Use the model below and fill it thanks to the information below.

```
#' @description ...    ①
#'
#' @param original_books_bis A tibble with two columns.
#' original_book$text is the lines of the text.
#' original_book$book is the part (e.g. chapter, different book...)
#' of the full text to which the lines belong
#' @return token_sentence A tibble with two columns.
#' token_sentence$sentence is each sentence of the text.
#' token_sentence$book is the part (e.g. chapter, different book...)
#' of the full text to which the sentence belong
#'
#' @import ...    ②
#' @examples
#' ## library(...)    ③
#' ## token_sentence <- tokenizer.sentence.i(original_books_bis)

tokenizer.sentence.i    ④ <- function(my.texte) {

...    ⑤

  return(token_sentence)
}
```

1 Describe the particularity of this function to divide the text into sentences. For example, which tokenization function of R has been used, from which package and with which parameters/arguments. It has to be just on one line because the description is automatically collected to be displayed in the app.

② and ③ Write the name of every package needed in your function.

④ Replace i by the number you used for the name of your file (tokenizer_sentence_i.R).

⑤ Write the code of your function. It has in input a tibble with two columns (original_books_bis).

- original_book$text is the lines of the text.
- original_book$book is the part (e.g. chapter, different book...) of the full text to which the lines belong

Which is the output of load_data_i.R functions.

The code you write has to split the text in sentences and specify the book each sentence belong. The output must be a tibble with two columns (token_sentence) :

- token_sentence$sentence is each sentence of the text.
- token_sentence$book is the part (e.g. chapter, different book...) of the full text to which the sentence belongs.

c. Tokenizer word

To add a way to tokenize words, go to <your_path>\Internship_NLP_CU\preprocessing\tokenizer_word. Add a file with tokenizer_word_i.R for name, with i the next number which has not already been used (needs to have no gap between the numbers). For instance if there already are tokenizer_word_1.R, tokenizer_word_2.R and tokenizer_word_3.R in the folder, you will name your file tokenizer_word_4.R. Use the model below and fill it thanks to the information below.

```
#' @description ...    ①
#'
#' @param original_books_bis A tibble with two columns.
#' original_book$text is the lines of the text.
#' original_book$book is the part (e.g. chapter, different book...)
#' of the full text to which the lines belong.
#' @param k A integer, the number of the sentence in the text.
#' @return token_word A tibble with three columns
#' token_word$word is each word of the sentence k, in the same order as in the sentence
#' token_word$sentence is the number of the sentence each wod belongs
#' token_word$book is the name of the book each word belongs
#'
#' @import ...    ②
#' @examples
#' ## library(...)    ③
#' ## token_word <- tokenizer.word.i(original_books_bis)

tokenizer.word.i  ④  <- function(my.texte) {

...    ⑤

  return(token_word)
}
```

1 Describe the particularity of this function to divide a sentence of the text into word. The whole text is divided into words by dividing each sentences of the text into words. For example, which tokenization function of R has been used, from which package and with which parameters/arguments. It has to be just on one line because the description is automatically collected to be displayed in the app.

2 and 3 Write the name of every package needed in your function.

4 Replace i by the number you used for the name of your file (tokenizer_word_i.R).

5 Write the code of your function. It has in input a tibble with two columns and just one row (token_sentence[k,]).

- token_sentence$sentence is the k-th sentence of the text.
- token_sentence$book is the part (e.g. chapter, different book...) of the full text to which this sentence belongs

Which is one line of the output of token_sentence_i.R functions.

The second input is k an integer, is the number of the sentence which has been selected.

The code you write has to split the selected sentence and specify the book and sentence each word belongs to. The ouput must be a tibble with three columns (token_word) :

- token_word$word is each word of the sentence k, in the same order as in the sentence
- token_word$sentence is the number of the sentence each word belongs
- token_word$book is the name of the book each word belongs

d. Normalization

To add a way to normalize words, go to <your_path>\Internship_NLP_CU\preprocessing\normalization. Add a file with normalization_i.R for name, with i the next number which has not already been used (needs to have no gap beetween the numbers). For instance if there already are normalization_1.R, normalization_2.R and normalization_3.R in the folder, you will name your file normalization_4.R. Use the model below and fill it thanks to the information below.

```
#' @description ...    ①
#'
#' @param token_word_freq A tibble with four colums.
#' token_sentence$word are the words of the text in alphabetical order occuring just one.
#' token_sentence$sentences is the list of numbers of sentences (line of the sentence in token
    _sentence) in which each word appear.
#' token_sentence$freq is the frequence each word appears in the text.
#' token_sentence$tf is the terme frequency of each word.
#' @return token_word_stem A tibble with four colums.
#' token_word_stem$word are the normalize form of words of the text in alphabetical order
    occuring just ones.
#' token_word_stem$sentences is the list of numbers of sentences (line of the sentence in
    token_sentence) in which each normalize word appear.
#' token_word_stem$freq is the frequence each normalize word appears in the text.
#'
#' @import ...    ②
#' @examples
#' ## library(...)    ③
#' ## token_word_stem <- normalization.i(token_word_freq)

normalization.i    ④ <- function(my.texte) {

...    ⑤

  return(token_word_stem)
}
```

① Describe the particularity of this function to transforming word into a single canonical form (car, cars, car's, cars' → car). For example, which normalization function of R has been used, from which package and with which parameters/arguments. It has to be just on one line because the description is automatically collected to be displayed in the app.

② and ③ Write the name of every package needed in your function.

④ Replace i by the number you used for the name of your file (normalization_i.R).

⑤ Write the code of your function. It has in input a tibble with four columns (token_sentence):

- token_sentence$word are the words of the text in alphabetical order and are not repeated.
- token_sentence$sentences is the list of numbers corresponding to the sentences (row number of the sentence in $token_sentence$) in which each word appears.
- token_sentence$freq is the frequency at which each word appears in the text.
- token_sentence$tf is the term frequency of each word.

The code you write has to normalize a list of words, associated with their right frequency and a list of sentences each word belongs to. The output must be a tibble with three columns (token_word_stem):

- token_word_stem$word are the normalize forms of words of the text in alphabetical order and are not repeated.
- token_word_stem$sentences is the list of numbers corresponding to the sentences (row number of the sentence in token_sentence) in which each normalized word appears.
- token_word_stem$freq is the frequency each normalized word appears in the text.

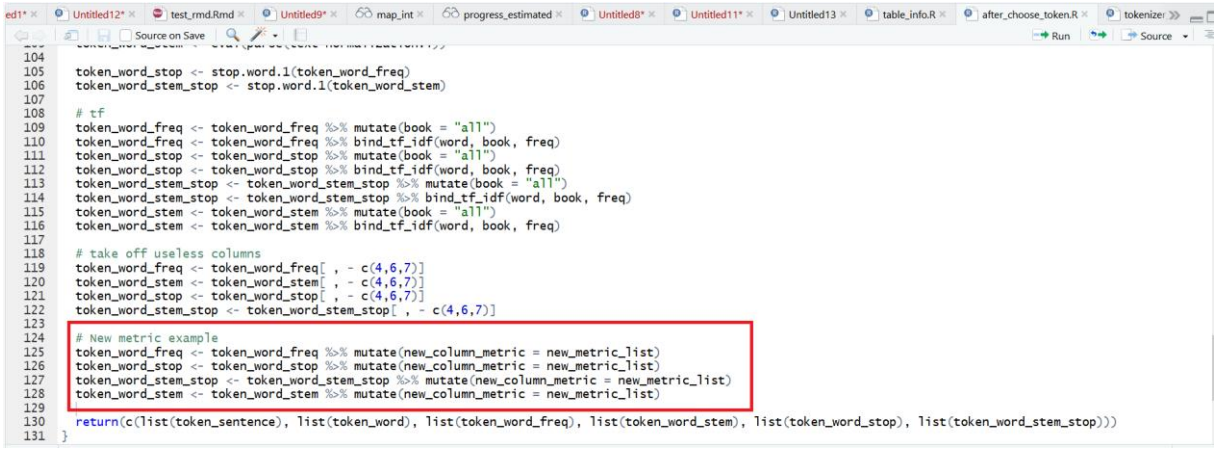
e. Add a metric

In the 'Overview' screen of the analysis part, it is possible to add a metric. Currently, there are two metrics: frequency and term frequency. However, they are not the only one a researcher would want to analyze a text. That is why it is possible to add a metric in this screen. In order to do it, there are three files that need changes. Let's say you want to add 'New Metric Example' as a new metric.

The first file to change is after_choose_token.R. This file creates the data that will be used in the plot. In this file, you need to add a column with the results of the metric in the output. Here is what you need to add after line 122 and before the return:

```
# New metric example
 token_word_freq <- token_word_freq %>% mutate(new_column_metric = new_metric_list)
 token_word_stop <- token_word_stop %>% mutate(new_column_metric = new_metric_list)
 token_word_stem_stop <- token_word_stem_stop %>% mutate(new_column_metric = new_metric_list)
 token_word_stem <- token_word_stem %>% mutate(new_column_metric = new_metric_list)
```

Where new_column_metric will be the name of the column with the values of the metric and new_metric_list is the list of values of the metric, new_metric_list[i] is the value of the metric for the word i. Here is what it looks like in the real code:



The second file to change is ui.R. This file takes care of the layout of the app. In it, go to line 159, and add the option you want, 'New Metric Example', after 'Term Frequency'. It looks like just below here:

The last file to change is server.R. This file is the backend of the app. It does all the calculation using different files and tells the app exactly what to do. In order to change the metric, go to line 614 and add two else if conditions. These conditions will be the copy of line 618 to 620 and line 627 to 629. You need to break a line after line 620 and write:

```
else if(input$choice=='New Metric Example'){

    plot_ly(original_books_tokenized_freq_shared(), x = ~rowname, y = ~new_column_metric, key = ~key(), type = 'scatter', mode='lines+markers',  marker = list(color = 'blue', opacity=2))%>%layout(title = 'New Metric Example according to the word', xaxis = list(title ='Word'), yaxis =list(title ='New Metric Example'), titlefont = 'arial', showlegend = FALSE)%>% highlight("plotly_selected", 'plotly_deselect', defaultValues = s, color = I('green'))

  }
```

Then go to line 629, break a line and write:

```
else if(input$choice=='New Metric Example'){

    plot_ly(original_books_tokenized_freq(), x = ~rowname, y = ~new_column_metric, key = ~key(), type = 'scatter', mode='lines+markers',  marker = list(color = 'blue', opacity=2))%>%layout(title = New Metric Example according to the word', xaxis = list(title ='Word'), yaxis =list(title ='New Metric Example'), titlefont = 'arial', showlegend = FALSE)

  }
```

f. Add information to the table info

In the 'Details on Demand' screen of the analysis part, it is possible to add a line to the table which summarizes all the numbers related to the text. You could for example add the number of negative

sentences or the number of passive sentences. In order to do it, there are one file that need changes. Let's say you want to add 'New Info Example' as a new information to put in the table. \newline

The file to change is Internship_NLP_CU-master\backend_analysis\table_info.R. In this file, you just need to add a line to the table_info. For that you can add the code below to the function table_info.R:

```
ligne <- list("New Info Example", "Symboles for New Info Example", Number related to the New Info Example)

names(ligne) <- c("Variables","Symboles", "Values")

table_info <- dplyr::bind_rows(table_info,ligne)
```

IV. Identified bugs

a. Installation

1. OpenNLP/RJava package problem

Error in loadNamespace(name) : there is no package called 'RJava'

This error concerns the OpenNLP package that uses the package RJava that need Java to be installed. On PC, if you install Java with the same type as R version, it will work (32 bits or 64 bits for both). On MAC, it didn't seem to work either way. Indeed, having Java is not enough, a JVM of Java must apparently be installed as well as Java and it seemed hard to do.

2. TinyText package problem

This error appears when you have the MCSafee antivirus active on your computer. MCSafee will cancel the installation of TinyText package. It thinks the installation contains a Trojan Horse. This package enables you to download your reports in PDF. If you don't have it, HTML and Word options will still work. If you want to have the PDF option, just disable MCSafee during the installation of this package or during the whole installation of the packages. There is not any malware in the installation, just packages from CRAN-R. If you want to be safe without this problem, you can use antiviruses such as Avira which are fine with the installation.

b. Application

1. Not all words display on wordcloud

On the wordcloud of the "filter" screen of the analysis, sometimes all the selected do not appear. This can be an annnoying problem. The problem is that in some conditions, the wordcloud does not have all the space to put the selected words. The issue is that there is no message to warn the user that words were not displayed and which word were not. This is an issue that can be found at this link. An idea to fix this bug if you really need a word and it never display is firstly to try to reselect the words you want and see if the wordcloud display the one you want. You can do it several times. If it keeps happening, you can change a parameter in the code that should help displaying more words. The file to change is \Internship_NLP_CU-master\app\server.R. Go to line 714:

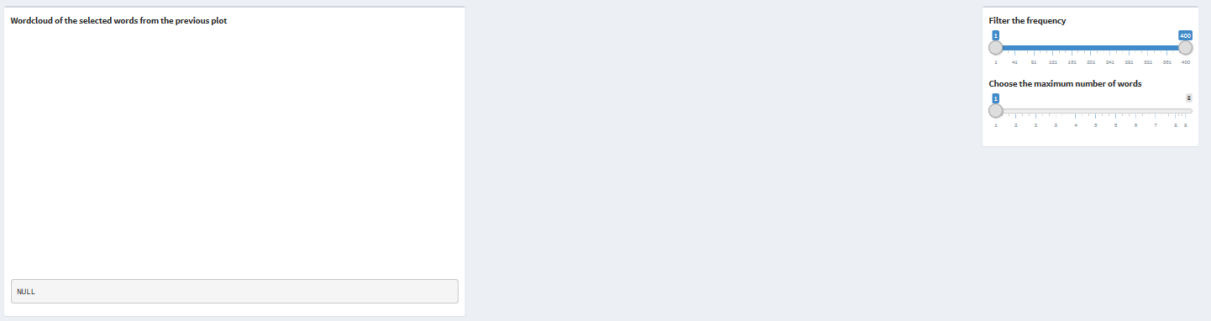
```
#Creating the wordcloud and making it reactive to change in the input values
 output$wordcloud  <- renderWordcloud2(wordcloud2(data= filter_d(), size = 0.1))
```

The parameter to play with is size. Try to reduce the size until your word appear, and then do a tradeoff between the size of the wordcloud and the words displayed on it.

Files affected:

- \Internship_NLP_CU-master\app\server.R

2. One word word cloud



In the "Filter" screen of the analysis, when there is only one word selected, no word appears. This problem can appear either by selecting only one word in the plot in the "Overview" screen, either by putting the slider bar of the maximum of words to 1. There is no fix of this bug. It is not a big problem because people usually don't need to have one word wordclouds.

Files affected:

- \Internship_NLP_CU-master\app\server.R

3. Wordcloud selection doesn't work

In some browsers, it is impossible to select a word in the wordcloud and so impossible to finish the analysis. In this case, you can either choose to run the app in R studio but it will be more complicated to download the report. To do this, go in the main.R file and change TRUE in FALSE in line 66 like this:

```
runApp(my_path, launch.browser = FALSE)
```

The known supported web browsers are:

- Microsoft Edge 10
- Google Chrome

The known non supported web browsers are:

- Mozilla Firefox

Files affected:

- \Internship_NLP_CU-master\app\ui.R

4. Plot selection doesn't work

Sometimes, it is impossible to select the data on the plot (in the overviews screen of Pre-Processing and Analysis). This can have two reasons, either the data is very big and the selection is very slow but by being very patient, it should work, either this doesn't work at all (big or small data). In this last case, you can try the same trick as in …… !!!!!!!!!!!!!!!!!!!!!!!! faire renvoi avec paragraphes bien !!! ,it should work.

5. Heaps law regression doesn't work

Error in int_abline: 'a' et 'b' must be finited

This error appears when there is not enough data and the regression is vertical line and so it is impossible for the computer to do a linear regression.

Files affected:

- \Internship_NLP_CU-master\backend_analysis\heaps_law.R

6. Problem RunApp

Error in runApp(lien, launch.browser = TRUE) :

could not find function "runApp"

When this error appears, that means that Shiny package is either not installed either not loaded. To solve this bug, write the following lines in the R console:

install.packages("shiny")

library("shiny")

c. Report

1. Wordcloud warning

Warning in wordcloud(dswf$ds.word, dswf$ds.freq, random.order = FALSE,

rot.per = 0.4, : cancer could not be fit on page. It will not be

plotted.

In the reports, sometimes all the words can't fit into the wordclouds. In that case, there will be a warning like the one above for each word that cannot fit. It is not nice visually but very useful to see which word is missing.

Files affected:

- \Internship_NLP_CU-master\app\report.Rmd

2. Backslash PDF

! Undefined control sequence.

l.1813 \n

\n\#Driven_Politics;;``,...

Warning: Error in : Failed to compile C:\Users\Projet\AppData\Local\Temp\RtmpYBqmQe\file27bc11ea289b.tex. See file27bc11ea289b.log for more info.

When the selected sentences in the "Word in Context" screen have backslahses in it, the PDF option for the report doesn't work. LaTeX thinks it is a command for him and so doesn't understand it. If you still want a PDF, do the Word option, then export the document to PDF.

Files affected:

- \Internship_NLP_CU-master\app\report.Rmd

d. Warnings

1. Boxplot warning

Warning: 'box' objects don't have these attributes: 'mode'

Valid attributes include:

'type', 'visible', 'showlegend', 'legendgroup', 'opacity', 'name', 'uid', 'ids', 'customdata', 'hoverinfo', 'hoverlabel', 'stream', 'y', 'x', 'x0', 'y0', 'whiskerwidth', 'boxpoints', 'boxmean', 'jitter', 'pointpos', 'orientation', 'marker', 'line', 'fillcolor', 'xaxis', 'yaxis', 'idssrc', 'customdatasrc', 'hoverinfosrc', 'ysrc', 'xsrc', 'key', 'set', 'frame', 'transforms', '_isNestedKey', '_isSimpleKey', '_isGraticule'

This happens when tracing the boxplots. This a warning with no solution currently, see the issue 994 of plotly at this link. It is because the plot_ly function has a bug when asked to trace a boxplot and a scatter of points in the same plot.

Files affected:

- \Internship_NLP_CU-master\app\server.R

2. RColorBrewer warning

Warning in RColorBrewer::brewer.pal(N, "Set2") :

minimal value for n is 3, returning requested palette with 3 different levels

This warning occurs when there are less than three different books in the data. The warning and a way to fix it have been issued here.

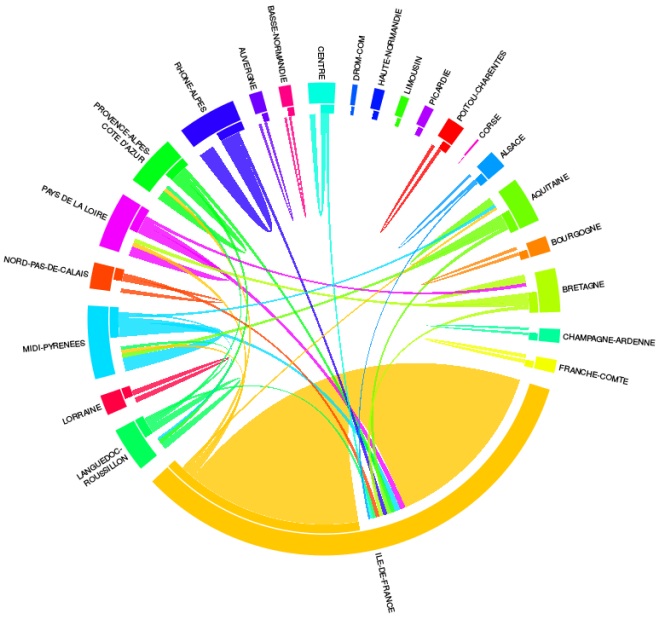
Files affected:

- \Internship_NLP_CU-master\app\server.R

V. Improvements

This application is a work in progress, it is supposed to be pursued and improved. Here are some ideas we did not have the time to do in order to improve the application:

- We could do a grammar analysis of each sentences in order to know the grammatical function of each word.
- We could have used a Chord Diagram in order to visualize the relations between word. The links could represent words that are in the same sentence or words which have a special grammatical relation (for example linking two words which have a subject object relation, or a noun/adjective one). The size of the link could be proportional to the number of time this relation is present in the text. The same thing could be represented with a simple graph.



- Some interesting numbers could be added to the summary table in the "details and demand" part of the app. For example, the number of passive sentences, or the number or negative sentences.
- Put some check box in the 'Word in context' part to filter just some sentences. For instance, print just the negative sentences, the passive sentences, or the negative passive sentences.
- Add a metric in the overview part in order to do a more in-depth statistical analysis.
- Add some state of the art word tokenizations, sentence tokenizations, normalizations, stop word lists, to have even more relevant boxplots.
- Offer to do only the statistical analysis part by taking an already tokenized text as an input.
- If it is appropriate with the data, put the app on an external server so that you can use it without having to install any package. We did not do it because we used data with restricted access which can only be upload on a local computer.

Contact

If you have any question about the app, troubles to install/use it or would like to report bugs, please feel free to contact us.

Colette Voisembert c.voisembert@gmail.com

Leon Migus leonmigus@gmail.com