UNIVERSITÉ DE LORRAINE

IDMC

Data Science Project Report

Omar Manil Bendali, Mayank Mishra, Samir Ferroum And Juba Ait Abdelmalek

M1- Natural Language Processing

May 15, 2023

Introduction

This report discusses the methods and results obtained for the project. We use the internet to extract our data and use Python programming to answer basic questions about the data in the context of Natural Language Processing.

Part 1

This part of the project deals with the extraction of the biographies of two categories of people. This data is then analyzed to look for text features specific to each category of people. Finally, we perform a classification task (classifying a randomly chosen text from the data into its category) on all of the data.

Data Collection

We choose the following two categories of people: mathematicians and painters. For data collection, we use the most readily available web data i.e. Wikipedia. We carefully choose wiki endpoints from where we could extract a large amount of data for the two categories of people easily. Using these endpoints we extract data files in a lexicographical order and tag them with their categories.

Data Analysis

Starting from the scraped and annotated raw corpus, we perform sentence segmentation and tokenization using spacy. We then normalize our data by removing punctuation, and additional spaces. To assess the effect of normalization and stop words removal, we create 2 versions of our data:

- norm (lowercasing + punctuation removal + non Latin words removal + stopwords removal)
- norm_with_stops (lowercasing + punctuation removal + non Latin words removal)

In all our subsequent plots for part 1, labels 0 and 1 represent mathematicians and painters respectively.

Sentences

We compute the average sentence length for each category and present it using a barplot and a boxplot.

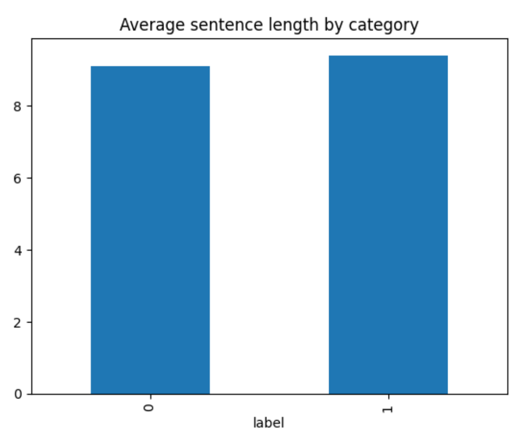

Figure 1:

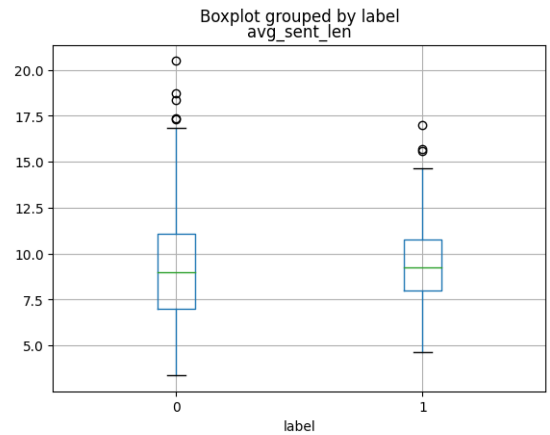

Figure 2:

We observe that the average sentence length for painters is nearly the same as the mathematicians. The boxplot displays the average sentence length distribution per category per biography. We notice a few outliers for both categories (i.e. biographies that contain a significantly bigger average sentence length than the rest of the articles).

The following table presents the min-max-average values of sentence lengths per category:

Category	**Min**	**Avg**	**Max**
Mathematicians	1	9.1	131
Painters	1	9.4	87

Table 1:

Token Occurrence

We count the token occurrences per sentence for each category and obtain the following results :

Category	**Min**	**Avg**	**Max**
Mathematicians	1	1.02	15
Painters	1	1.02	13

Table 2:

We see that the token "academy" appeared the most i.e., 15 times in a sentence for the category mathematicians, and the token "Pakistan" appeared the most i.e., 13 times in a sentence for the category painters.

The following table presents the total number of tokens per category, with and without stop words:

Category	**Stop words and no normalization**	**No stop words and normalization**	**Stop words and normalization**
Mathematicians	329,953	159,091	267,425
Painters	190,117	91,368	158,677
Total	520,070	250,459	426,102

Table 3:

Below we provide a stacked bar chart for the above table.

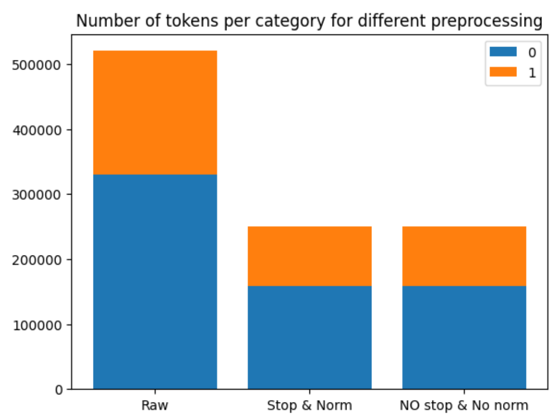

Figure 3:

We notice that after normalizing we are able to reduce the number of tokens by half. We also compute the 10 most common tokens per category before and after removing the stop words, and present the results in the following bar charts:

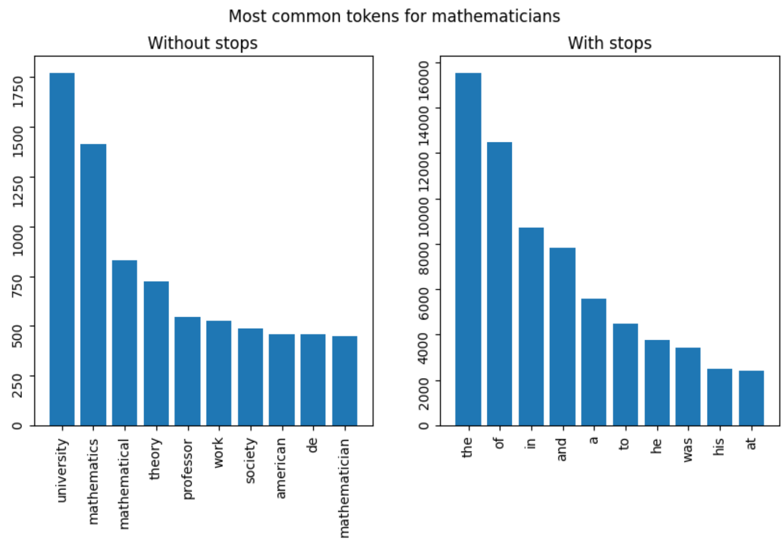

Figure 4:

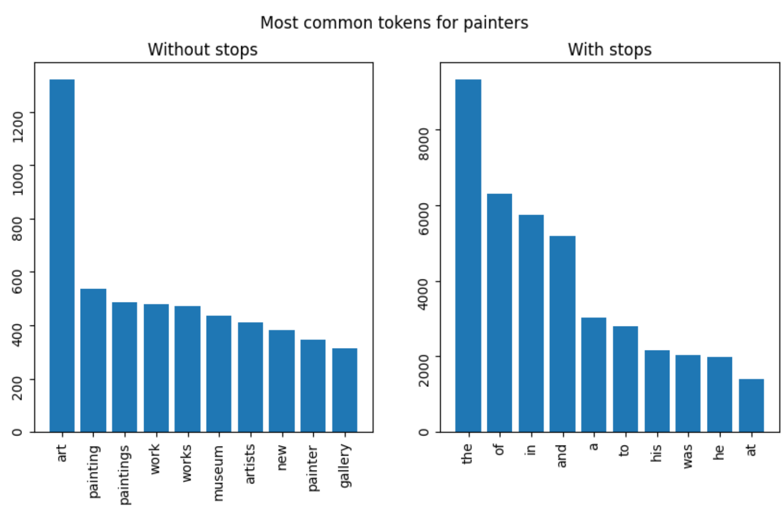

Figure 5:

We notice that the most common tokens when we keep stop words are the stop words themselves. For example, in the mathematicians' category, the most common stop word is the token "the" (8000) while the most common word which is not a stop word is "university" (1750). This significant gap in token occurrence can bias our further processing since the stop words may outweigh other tokens. That is why we argue that removing the stop words is recommended for further processing.

Vocabulary

To visualize the difference in vocabulary between the categories, we display the word cloud of each of them separately.

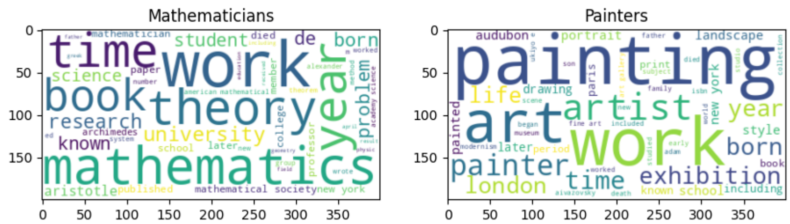

Figure 6:

POS Tags/Named Entities

Since we want to visualize what are the most common POS tags for each category, we produce the following bar charts.

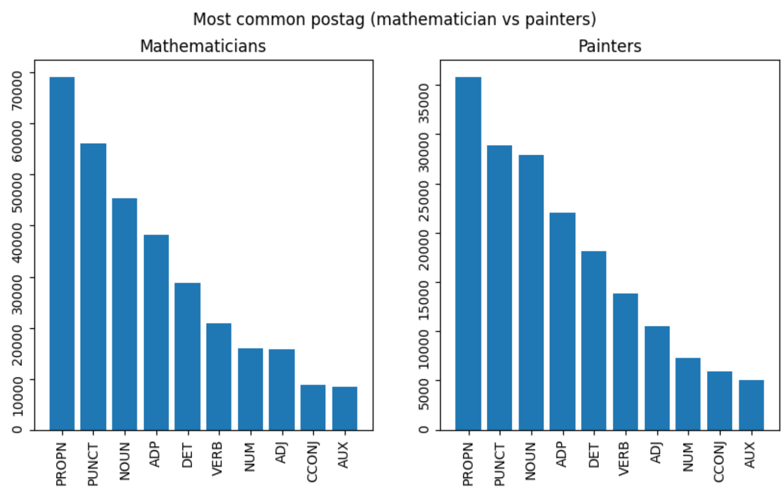

Figure 7:

We notice that the most common POS tags are the same for both categories. Furthermore, we explore the most commonly used named entities between the two categories.

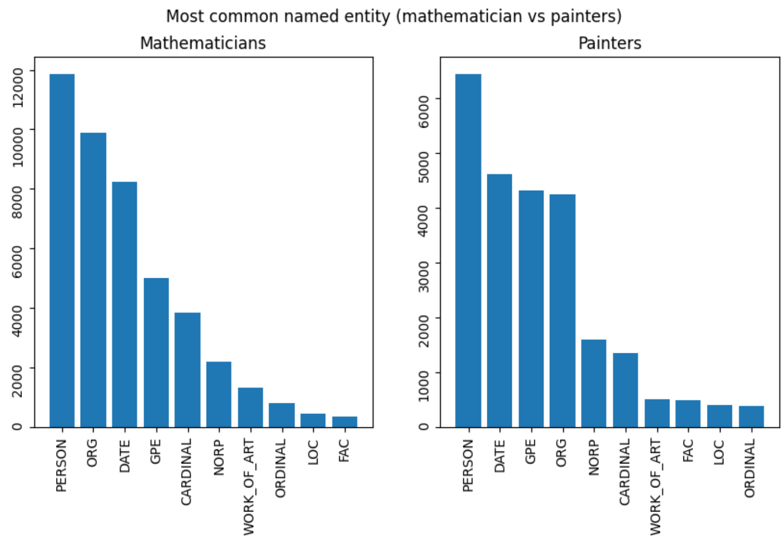

Figure 8:

We observe that the named entity PERSON is the most common for both categories. However, the second and third most commonly used named entities are different. For mathematicians, the second most common named entity is ORG (Companies, agencies, institutions), and for painters, it is DATE. Later, we evaluate the most common couple (token, NE).

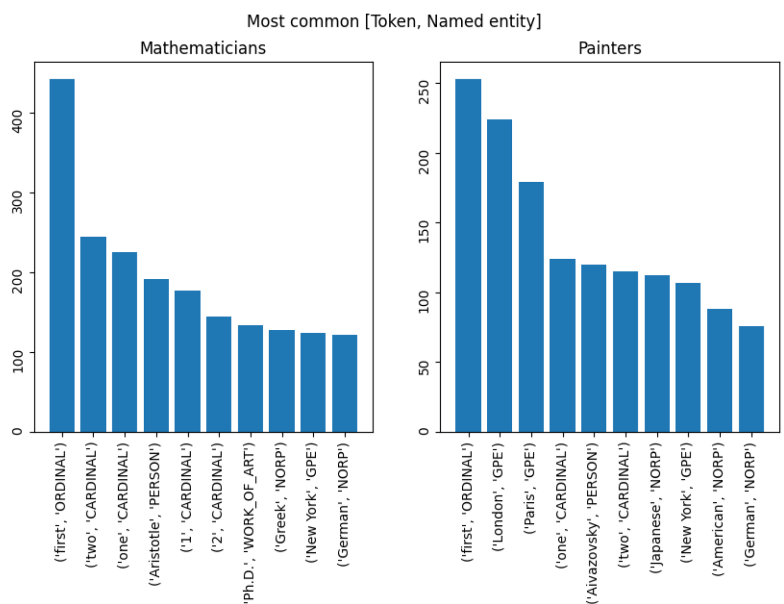

Figure 9:

The first most common is the ORDINALS (first, second, etc.). For mathematicians, the most cited PERSON is "Aristotle", while for painters it is "Aivazowsky".

Classification

Before applying any classification model on our data, we first evaluate its class distribution. The following table displays how many mathematicians and painters biographies our corpus contains.

Label	Number of articles	Number of articles (under-sampled)
Mathematicians	460	185
Painters	185	185

Table 4:

We notice that the classes are highly unbalanced, and this could affect the performance of our classifier. Hence, we choose to under-sample the category mathematicians to make the number of mathematicians' biographies equal to the painters'.

For our classification task, we define a pipeline that will be used throughout our task:

- TF-IDF vectorization
- Classifier

We choose to perform hyperparameter tuning on our data using two classifiers:

- Random Forest
- Multinomial Naive Bayes

We use Grid search and include the classifier as a parameter in the search space, to avoid running a grid search for each classifier.
We also iterate over different types of pre-processing, namely: raw text, normalized text (norm), normalized with stop words (norm_with_stops), and apply to each the grid search defined. We save the grid search results and compile them to see what are the best hyperparameters. Next, we shuffle and split our data into training (80%) and test data (20%) and retrain classifiers using the best hyperparameters obtained previously.

Remark: In this approach, we choose to explore hyperparameters space using the whole data, but it is important to note that we do not save the best models but only the hyperparameters values, and only these values are fed to our classifiers (i.e the model have not been fitted on the train nor test data before). After training our classifiers, we test them on test data and compile their performances (f-scores) and confusion matrix. We obtain the following results for the top 5 models:

F1_grid	std_F1_grid	process	model	f-score
0.984468	0.015356	text	RandomForest	0.972243
0.986999	0.014612	norm_text	RandomForest	0.972243
0.986999	0.014612	norm_text	RandomForest	0.972243
0.981134	0.019492	norm_stops_text	MultinomialNB	0.972243
0.978782	0.019235	text	MultinomialNB	0.972243
0.986999	0.014612	norm_text	RandomForest	0.972243

Table 5:

We notice that on the test data all models perform equally well, and they are all able to classify between mathematicians and painters. This can be explained by the fact the vocabularies of the two categories are far from each other.
We also notice that the best models all have the same confusion matrix:

		Predicted label	
		Mathematician	Painter
True label	Mathematician	30	**1**
	Painter	**1**	42

If we explore what is the content of these misclassified texts we notice that:

- 1 false positive: misclassified because of an issue with data extraction, the scraped page by Wikipedia library is of General Sir John Miller Adye (an artist) and should have been of John Adye (a mathematician)
- 1 false negative: also misclassified because of a data extraction issue, the Wikipedia library is returned the page content of the given name Akimitsu

These results emphasize on the importance of validating the data collected before using it for any classification task.

Part 2

Data Collection, Sentence Segmentation, and Tokenization

Data Collection and Sentence Segmentation

For the second part of this project, we fetched random topics using Wikipedia API and extracted their summaries. This data is then utilized for sentence segmentation. To observe the differences in sentence segmentation outputs of the NLTK and SPACY libraries, we built a Python function that accepts a string input, that is a path of the folder which contains the random articles, and structures a desired output into a Pandas data frame. The data frame contains a column for the original text, a column for the shared sentences, a column for sentences unique only to NLTK, a column for sentences unique only to SPACY, and two other columns, in which each column contains the full segmentation of SPACY and NLTK. We argue that this helps to establish a better descriptive analysis of the task at hand and to see where both libraries may have handled the data differently.

	original_text	shared_sentences	unique_to_spacy	unique_to_nltk	nltk_segmentation	spacy_segmentation
0	Six judges of the International Criminal Court...	Six judges of the International Criminal Court...	If the creditor breaches the accord, then the ...	If the creditor breaches the accord, then the ...	Six judges of the International Criminal Court...	Six judges of the International Criminal Court...
1	NaN	The judges were elected for terms of nine year...	This temple, built by Chola emperor Rajaraja I...	This temple, built by Chola emperor Rajaraja I...	The judges were elected for terms of nine year...	The judges were elected for terms of nine year...
2	NaN	Accord and satisfaction is a contract law conc...	The Airavatesvarar temple is one among a clust...	It also reverentially displays Vaishnavism and...	Accord and satisfaction is a contract law conc...	Accord and satisfaction is a contract law conc...
3	NaN	It is one of the methods by which parties to a...	It also reverentially displays Vaishnavism and...	It was hypothesized that certain forms, such a...	It is one of the methods by which parties to a...	It is one of the methods by which parties to a...
4	NaN	The release is completed by the transfer of va...	The stone temple incorporates a chariot struct...	Though challenged in the 17th and 18th centuri...	The release is completed by the transfer of va...	The release is completed by the transfer of va...
...	...	...	...	...	...	...
1037	NaN	Voter turnout was only 25%.	NaN	NaN	NaN	Voter turnout was only 25%.
1038	NaN	The 20th Century Press Archives (German: Press...	NaN	NaN	NaN	The 20th Century Press Archives (German: Press...
1039	NaN	It originates from the Hamburg Kolonialinstitu...	NaN	NaN	NaN	It originates from the Hamburg Kolonialinstitu...
1040	NaN	In 2007 it was absorbed by the German National...	NaN	NaN	NaN	In 2007 it was absorbed by the German National...

Figure 10: Pandas data frame after sentence segmentation

Tokenization

The tokenization of the textual data was carried out using the Pandas data frame obtained previously. We first find the tokens for the unsegmented data and then compare this to the tokens in the segmented data.
For the unsegmented data, we find the vocabulary for both libraries, their shared vocabulary (i.e. tokens identified by both), and the set of unique tokens identified by each library. Finally, we also find the shared tokens for both libraries after sentence segmentation. We also do POS tagging simultaneously for this data to count the number of times both libraries assign the same tag in our shared tokens after sentence segmentation. The simultaneous tagging of data helps us to preserve the structure of each tokenized sentence and obtain better tagging quality in comparison with the case where we performed POS tagging on the common tokens only.

POS Tagging

Since we already performed the POS tagging during the previous step, we compute for each token of sharedTokensInSentences, the number of times and ratio when both Spacy and NLTK agreed on a tag. Moreover, we compute the frequency mapping for each tag for each library and display it in a confusion matrix.

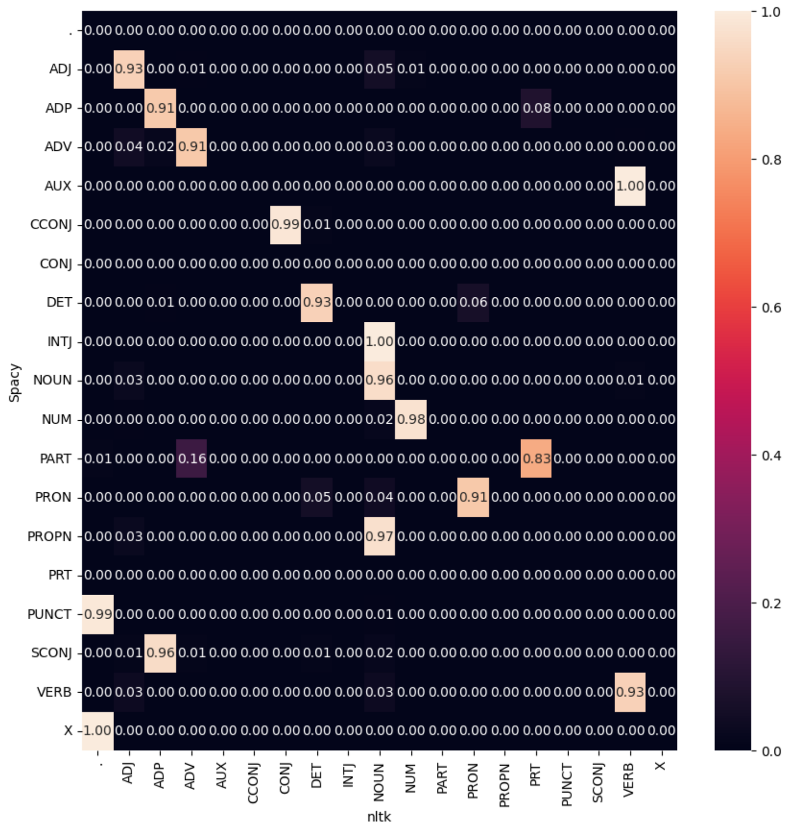

Figure 11: Confusion matrix from Spacy to NLTK

The most relevant observations from the obtained confusion matrix are :

- Spacy tags SCONJ (subordinating conjunction) are 96% of the time tagged as ADP (Adposition, cover term for prepositions and postpositions) by NLTK
- Spacy tags AUX (auxiliary) are 100% of the time tagged as VERB by NLTK.

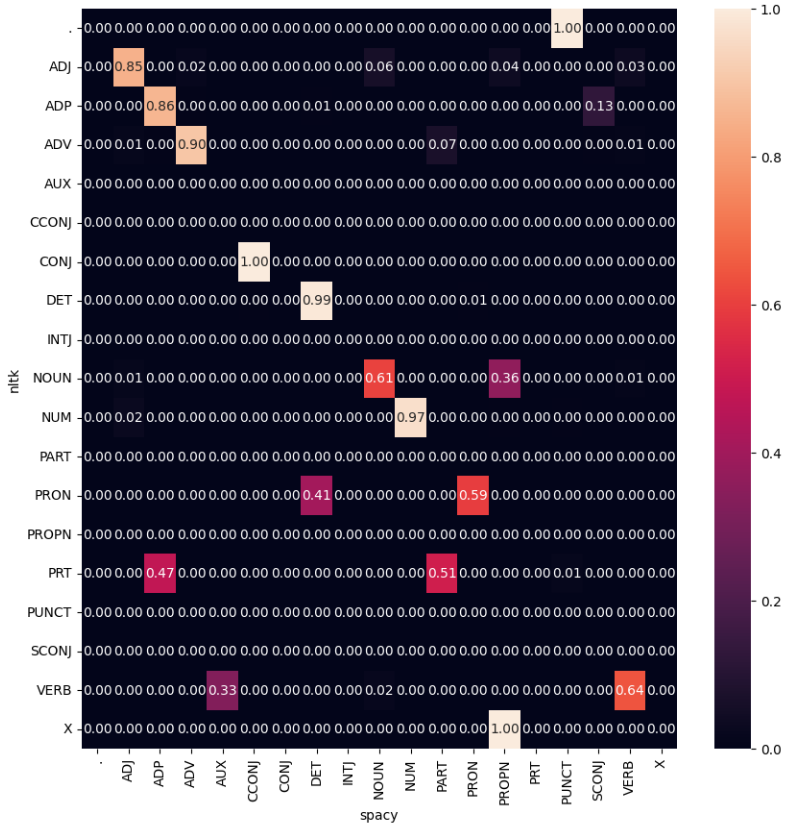

Figure 12: Confusion matrix from NLTK to Spacy

The most relevant observations from the obtained confusion matrix are:

- NLTK tags NOUN are 36% of the time tagged as PROPN by Spacy.
- NLTK tags PRON are 41% of the time tagged as DET and 59% as PRON.
- NLTK never recognized a PROPN.
- NLTK tags PRT (particle) are 47% of the time tagged as ADP by Spacy.

Dependency parsing

We perform a similar process for dependency parsing and generate a frequency mapping for each relation. Since NLTK doesn't provide a built-in parsing facility, we load the CoreNLP Stanford Parser in the library. Next, we add the relation NLTKNOREL to check the relations that were identified by Spacy and not by NLTK.

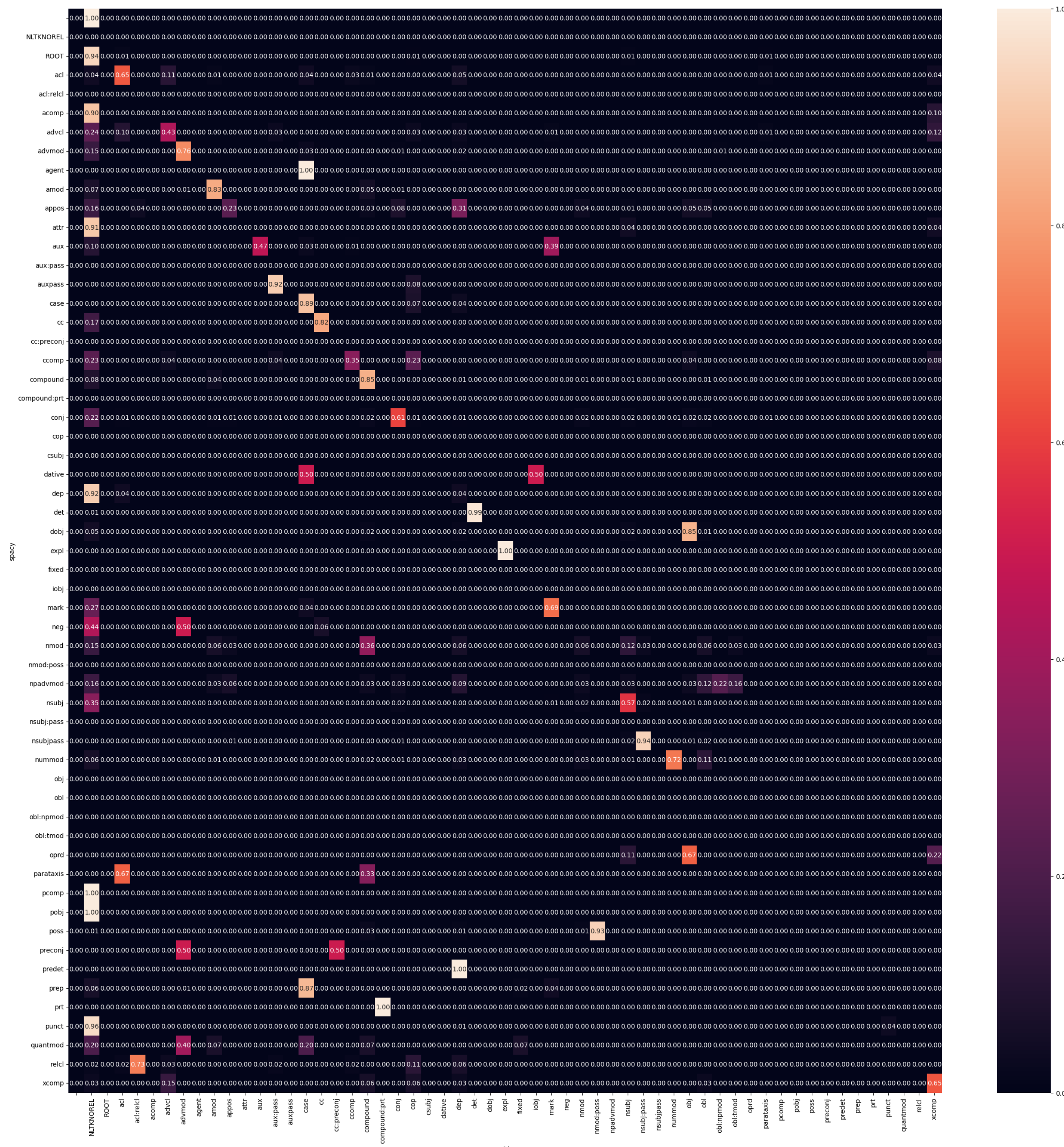
spacy
nltk
NLTKNOREL
ROOT
acl
acl:relcl
acomp
advcl
advmod
agent
amod
appos
attr
aux
aux:pass
auxpass
case
cc
cc:preconj
ccomp
compound
compound:prt
conj
cop
csubj
dative
dep
det
dobj
expl
fixed
iobj
mark
neg
nmod
nmod:poss
npadvmod
nsubj
nsubj:pass
nsubjpass
nummod
obj
obl
obl:npmod
obl:tmod
oprd
parataxis
pcomp
pobj
poss
preconj
predet
prep
prt
punct
quantmod
relcl
xcomp
1.0
0.8
0.6
0.4
0.2
0.0

Figure 13:

The most relevant observations from the obtained confusion matrix are:

- Spacy relation pcomp/pobj/punct/dep are never or very rarely ([90% – 100%]) identified by NLTK.
- Spacy doesn't tag the reflexive ROOT relation, unlike NLTK.
- Spacy relation agent is 100% of the time identified as a case by NLTK.

We later add the relation SPACYNOREL to define relations that were identified by NLTK and not by Spacy.

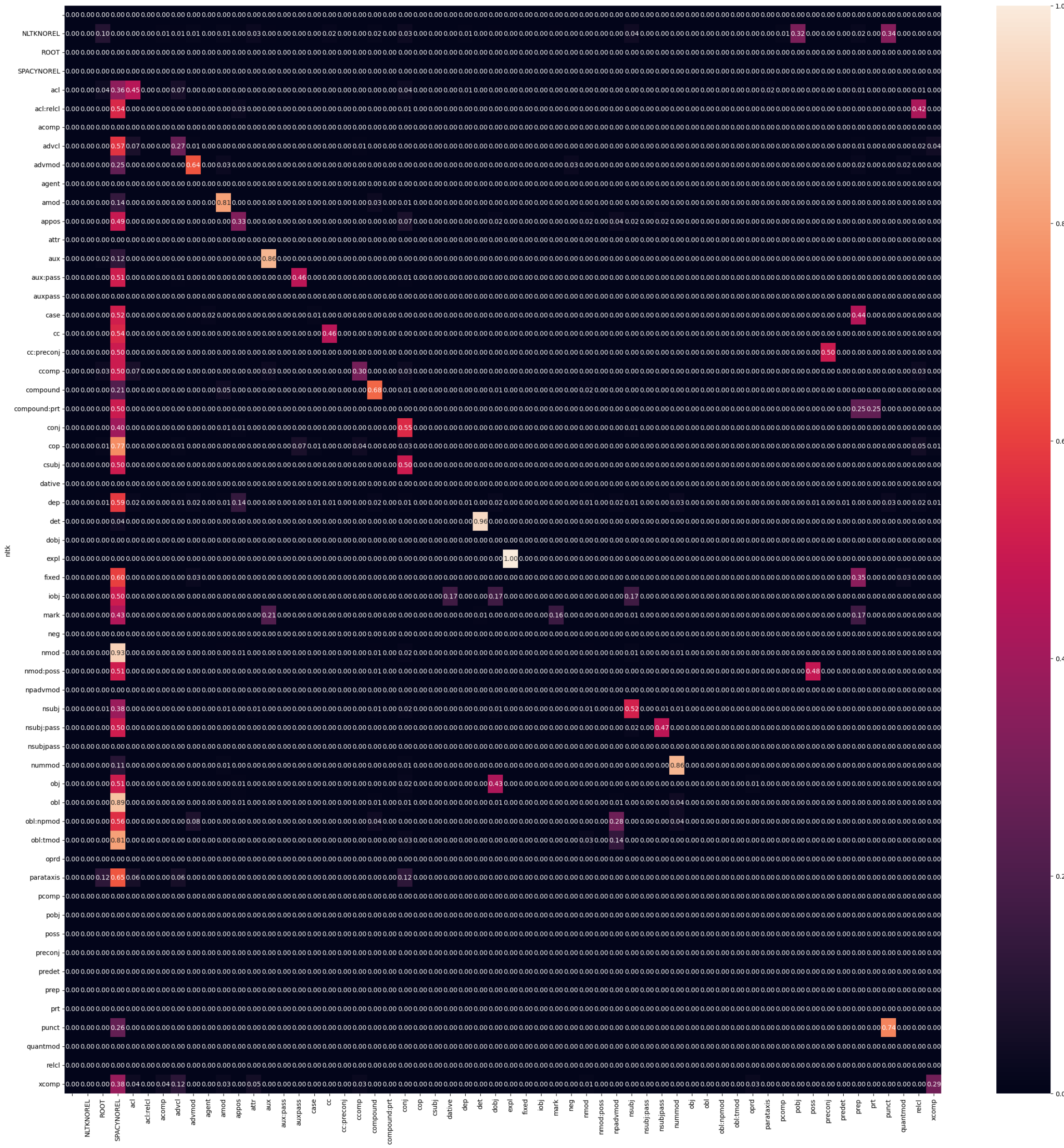

Figure 14:

The most relevant observations from the obtained confusion matrix are:

- NLTK relation nmod/obl/obl:tmod are rarely (> 80%) identified by Spacy.

Conclusion

In conclusion, we can say that the project was a clear illustration of how important is data pre-processing for descriptive analysis, more importantly so in the first part of the project. The focus on choosing an appropriate model for classification tasks is not as important as the quality of the data itself that will be fed to the model. 'Often in NLP, feeding a good text representation to an ordinary algorithm will get you much farther compared to applying a top-notch algorithm to an ordinary text representation.' In the second part, we see how quintessential it is to compare the performance of NLP libraries so that we tackle a specific NLP task with the appropriate library to obtain better results. Out of the scope of this project, we could use other well-pre-trained parsers. A great example of that is, 'XLM-RoBERTa' a state-of-the-art transformer-based model for multilingual NLP. Trankit, a tool built on top of XLM-RoBERTA that is already been compared to Stanza, UDpipe, etc., and achieved greater results from the simplest NLP tasks like POS-tagging to more complicated ones like dependency parsing.