JH Data Science Capstone : Language Modeling

Mohammed Ait-Oufkir

10 Januar, 2019

Executive Summary

The Goal of this NLP project is to predict which is the most likely word that comes next in a given context (history of preceeding words), based on a text corpus. This report describes the methodolgy and steps performed in order to acheive this task. A thorough exploratory analysis of the data helps understanding the distribution of the words and their relationships within the Corpora.

As expected from the text content, a cleaning step should be performed. The perfomance issue is adressed and sampling strategy is discussed. An exploration of the n-gram model is provided at the end of the analysis.

For the sake of clarity the source code producing this report Analysis is hidden. The complete source code and the sample data file can be found in the link here.

Data Analysis

The Corpus

The corpus is composed of the 3 categories of text :

- Tweets
- Blogs
- News

The data is provided in different languages (English, German, Finish and Russian). Only English Corpora is analyzed in this project, the same logic will be implemented (considering language specifities) for the other languages. The data set used in this project is SwiftKey data set and can be downloaded here

Visual inspection

A look into the data files shows us that :

The text contains some words/abreviations specific to tweets (specific dialect)

I'm coo... Jus at work hella tired r u ever in cali Dammnnnnn what a catch

it contains also some special characters :

I'm doing it!ðŸ‘Œ *â€œ: "The tragedy of life ...*

Usage of acronyms :

I have a 3.0 GPA, and for the NCAA (Clearing House)... The agencies assigned AAA ratings to securities ... a spokesman for AAA Mid-Atlantic...

usage of casual language, numbers and upper-case :

i'm not gonna be here much longer... :(Watcha know about half off 97 4337 Koeln Av, $97,500 gyeaaaaa. I know you gon do it. FRIENDLY SENIORS OF NORTH BERGEN

Some errors fonud in files : *readLines(conn) : line 167155 appears to contain an embedded nul*

No assumptions are given on the category of the text corpus which means that the text that comes out of (twitter, news and blogs) is a common language and belongs to no specific field such as Scientific corpus. The analysis of the words/ngrams and their frequencies will somehow give us an indication of the dominate words/ngrams that represent the language and thus how much it covers the language.

Statistics on the corpus

The following figures show how big is the corpus and gives a broad idea about the the amount of memory and the processing power needed to handle the data. we can clearly see that tweets has the most number of lines but the lines are shorter (no surprise knowing that tweets are restricted by number of characters), blogs contain longest line this is could be an indication about patterns in writing news and blogs. Considering the words we can see that Blogs then tweets have the most number of words (Total and Unique) but the lexical density of news $\frac{\sum W_{unique}}{\sum W_{All}}$ is higher. this may be explained by the richness of the news writing over repition in the case of blogs and tweets.

Source	Number.Lines	Length.Longest.Line	Length.Shortest.Line	Number.Words	Number.Unique.Words	Lexic
News	77259	5760	2	2643969	197858	0
Blogs	899288	40835	1	37334131	1103548	0
Tweets	2360148	213	2	30373543	1290170	0

Data Processing

After loading the corpus and as in any standrad text mining task we need to pass through a cleaning process which will ease relevant information extraction, even though cleaning may cause information loss (the information loss measurment is out of the scope of this capstone).

Cleaning the corpus and handling Profanity

After cleaning the document (removing stop words, Numbers , non alpha characters and unecessary spaces) we can see that the profane words (relative to the profane list used for this analysis) are `907568` out of `38359035` the total text so we can proceed to their removal. one question reamins do we need to remove the whole sentence that contain these words or only this words.

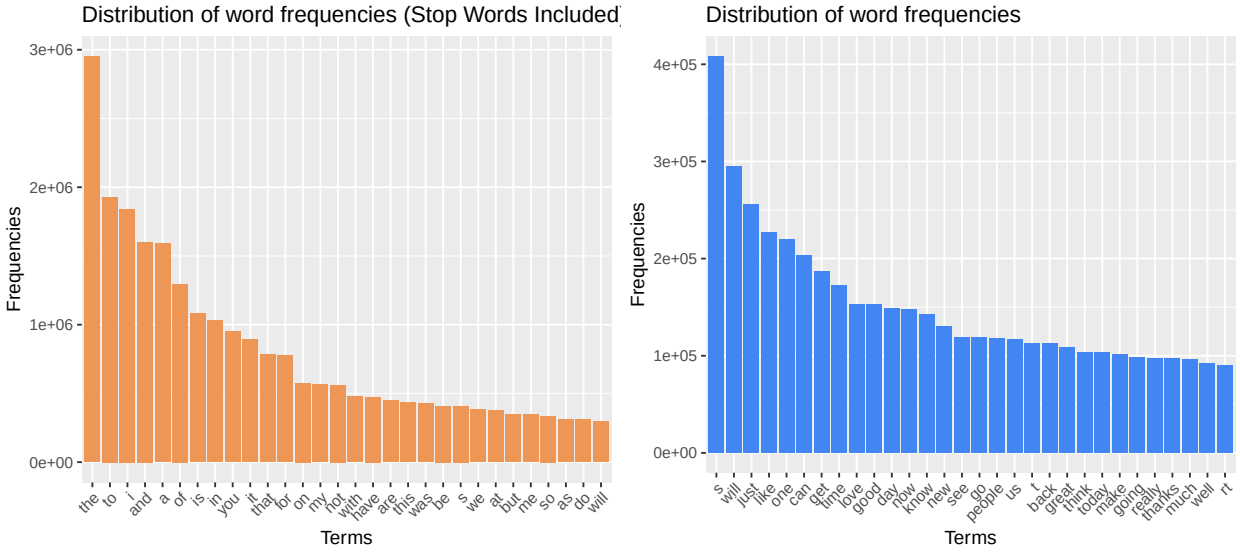
Analyzing word frequencies

We can see that the stop words are the most frequent but after removing stop words some single character words such as **s** and **t** emerges and so potentially the t after apostrophe did not get cleand out and need to be considered in the analysis.

	word	tot_occur
54989	s	408478
70212	will	295129
33046	just	256192
36506	like	226907
44840	one	219679
9202	can	203239
25120	get	187273
64537	time	172539
37312	love	153119
25831	good	152822

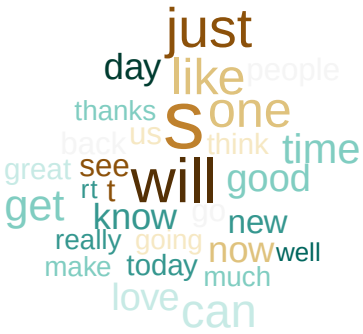
	word	tot_occur
64087	the	2950882
64813	to	1928269
30054	i	1840706
2102	and	1602199
3	a	1591667
44604	of	1296276
31920	is	1082572
30599	in	1031469
71528	you	952771
32016	it	891049

A bar plot shows visually the frequencies of these words.



Wordclouds

To highlight Visually the importance of words Wordcloud is a good choice.



Sampling and representativeness of the samples

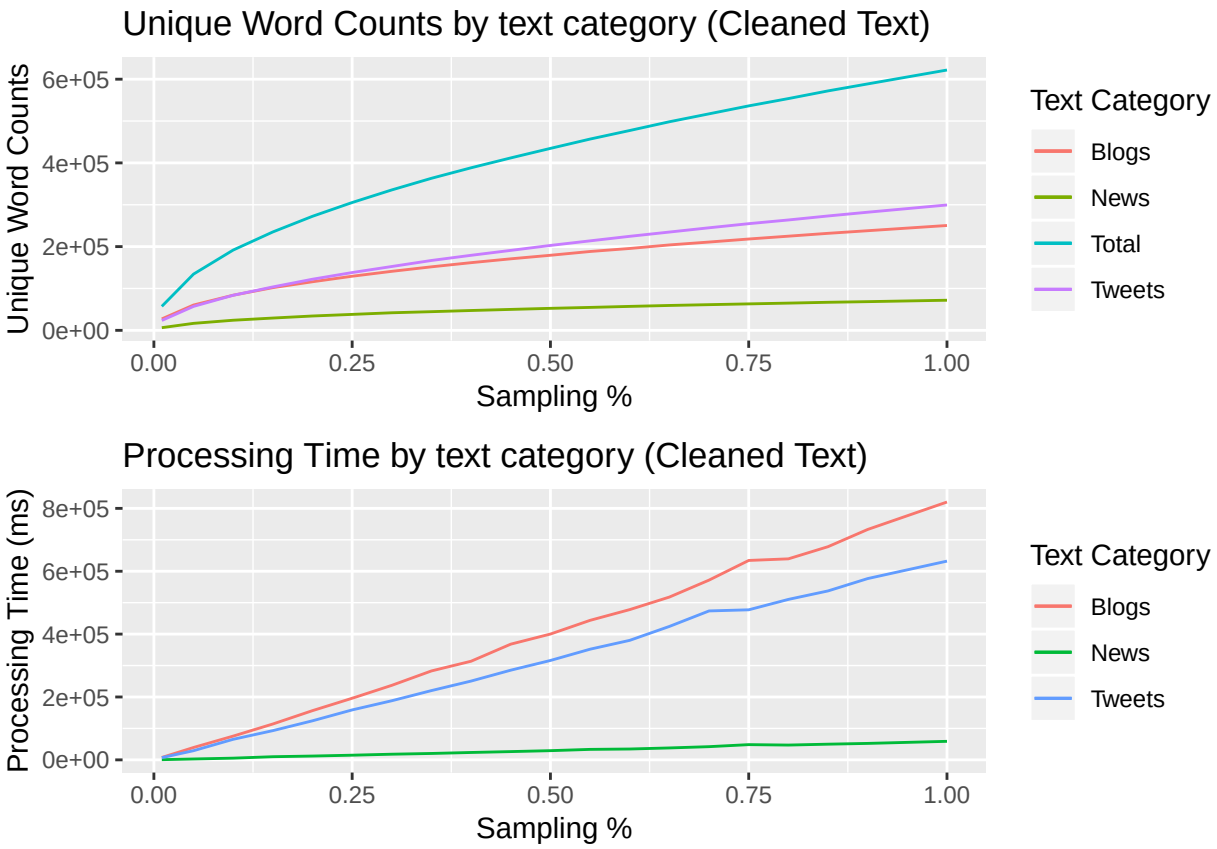
After creating a series of sample size from 1% to 90% we can observe that the number of unique words grow with the sample size, but this growth rate become smaller after a certain sample size.This confirms the observation from word frequencies that some words are predominant in the corpus.

The second graph shows that the processing time increase almost linearly as the sample size increases and this for the following configuration:

- Library used : TM
- Text cleaning and word frequency (unigram only)
- Machine configuration (CPU Intel Core i7 7th Gen 4 Cores, Memory : 16 GB)

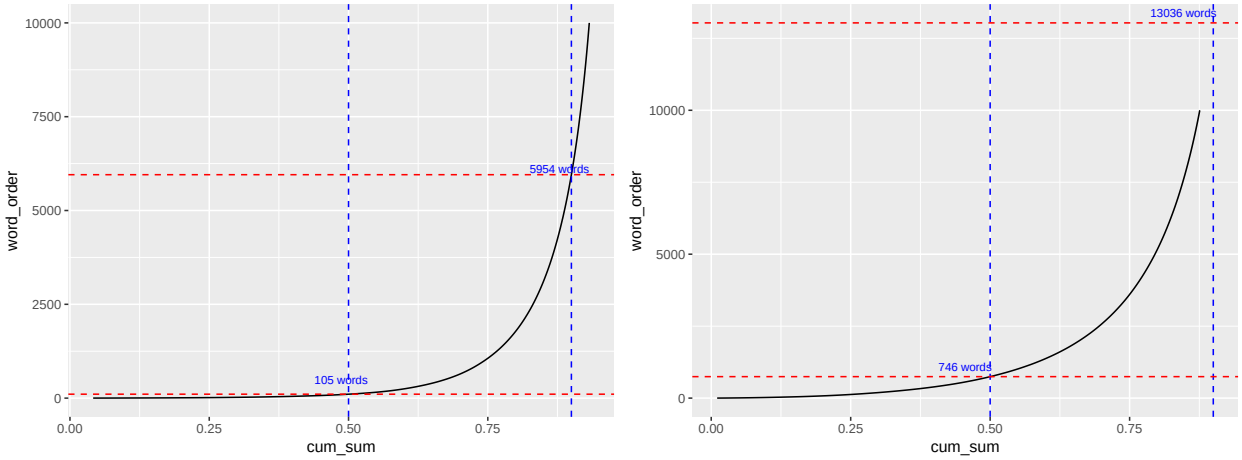
this gives us an indication about the time needed/complexity when we will generate bigrams and upper n-grams.

The conclusion is that sampling is a reasonable option. now we should investigate further wich sampling size and strategy to adopt in order to obtain a relevent result when runing word prediction.



Text coverage

Let see how many words are covering 50% and 90% of the corpus. For this we need to run a cumulative sum of the word frequencies and check at which number of words we hit a 50% of the corpus coverage and at which number we hit the 90% coverage.



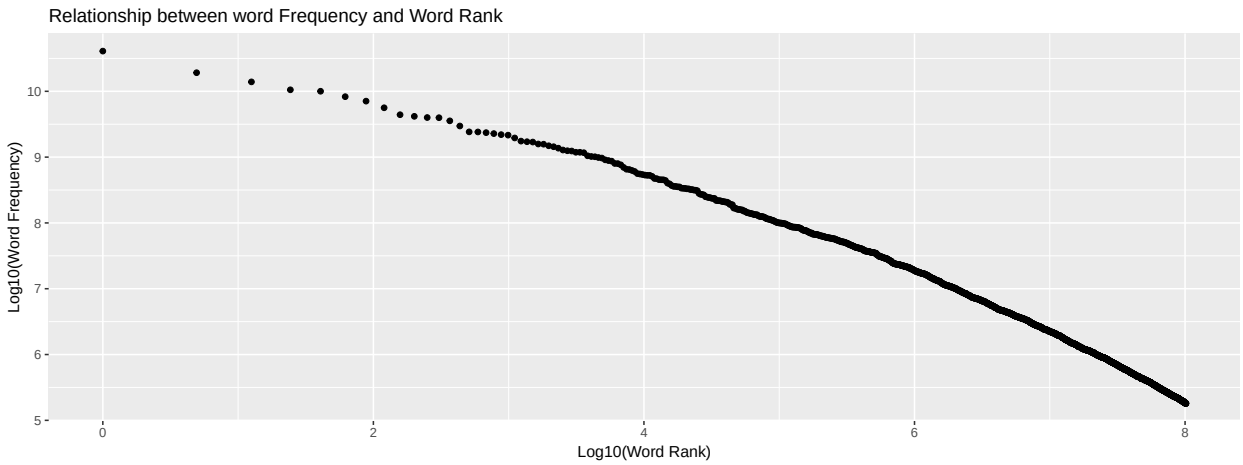
The number of words required for the text coverage can be summerized as follow

- The raw text (including stop words)

- 105 words to cover 50%
- 5954 words to cover 90%
- The cleaned text (no stop words and other processing methods as indicated above)
- 746 words to cover 50%
- 13036 words to cover 90%

to cover 50% we do not need a large dictionary in both cases but always less in the raw text that includes stop words (stop words will be used in the prediction model). on the other hand to raise the coverage till 75% we still do not need a large dictionary. combining the results from this graph and the previous graph (Unique word counts by text category) we can say that 10% sample size is a reasonable size for the project.

Another interesting property that we can observe in the data is the quasi-linearity of the relationship between the log(word Rank) and log(word frequency), this property known as zipf's law say's that the word frequency is inversly proportional to its rank.



Also in order to improve the quality of the sample we can envisage the following techniques:

- Stemming : will reduce the words to their stems and thus increase the frequency of the stem, the stem will count for any unseen word derived from the stem, but this will need a reconstruction of words from the stem if predicted (or propose all different words related to the stem in the solution).
- Smoothing techniques
- Transform the profanity into a normal language within the text (even if in the corpora there not a huge amount of profane words)

Foreign languages words evaluation

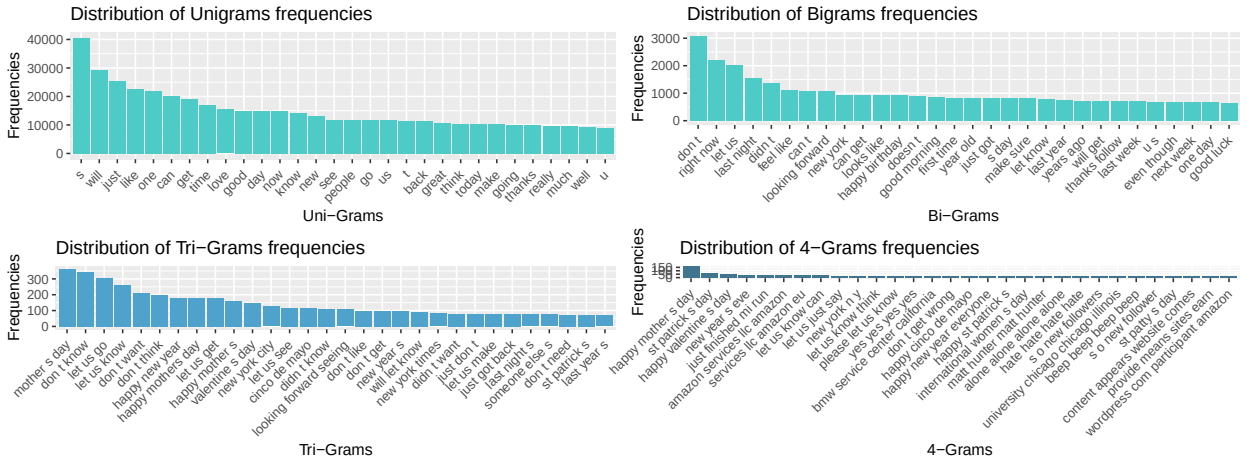
To spot words from a foreing language we can use one of the following techniques :

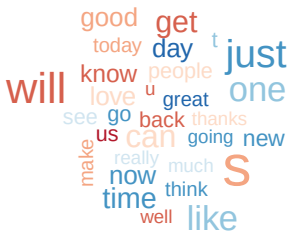
- The unicode could be a good option to spot words that are outside the unicode set for the language of interest.
- The usage of a fixed dictionary of the language of interest to spot words which did not belong to that dictionary.

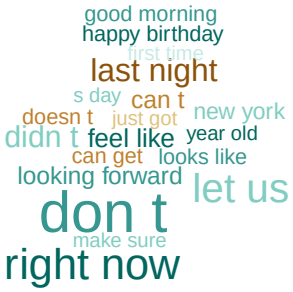
The analysis in this report consider only English corpora. but the same logic is valid for any language corpora.

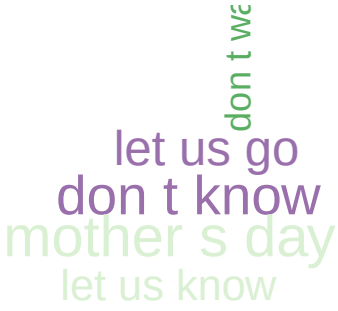
Generating the N-grams

The n-grams that are shown below are based on the 10% sample of the data. the ngram library is used for this step. further libraries will be explored from the performance stand point.









Conclusion & plan for the model implementation

The model that will be used in this analysis is the n-gram model. For example the prediction of the word **`now`** given the word **`right`** is calculated following the Maximum Likelihood Estimation (MLE) formula below:

$$P(now \mid right) = \frac{P(right\ \ now)}{P(right)}$$

* $P(now \mid right)$: This is what we wanted to calculate.

- $P(right\ \ now)$ The frequency of the bi-gram **`right now`** in the corpus.
- $P(right)$ The frequency of the uni-gram **`right`** in the corpus.

To avoid keeping track of all the possible ngrams in memory, we can rely only on a small set of ngrams e.g. $N \in \{2, 3, 4\}$ and use the Markov Property. in short the markov property tells that the probability of a new word depends only on the last word in the sentence and not the whole sentence.

So the idea is to approximate the next word from a set of N-grams ($N \in \{2, 3, 4\}$)

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n+N-1})$$

instead of calculating the complete history of previous words

$$\prod_{i=1}^{n} P(w_i|w_1^{i-1})$$

For the unseen words the Katz backoff smoothing is chosen.

The data is split into 2 parts a training and validation sets and the perplexity metric for the different ngrams will be adopted.

A shiny application will be developed to allow end user test the predictions.

Reference:

Profanity List

Number Of words in English Merriam-Webster

Number Of words in English Oxford

n-gram package

n-gram Model

Zipf's Law

zipf's Law Stanford

Contractions Management