Natural Language Processing Assignment 3 - Removing Stopwords from Corpus

Dr. Seba Susan

[Live Jupyter Notebook] [Project on GitHub]

20th August 2020

Anish Sachdeva
DTU/2K16/MC/013
Delhi Technological University

Index

Introduction *[See Notebook]*

Normally the following steps are very commonly and ubiquitously implemented as part of the data preprocessing pipeline in any NLP project:

1. Converting the entire corpus into a common case (either lowercase or uppercase)
2. Extracting words/tokens from the corpus
3. Removing all Punctuations from the selected tokens and retaining only alphanumeric quantities
4. Removing all Stopwords from the extracted tokens.
5. Stemming the Token into its root with any Stemming Algorithm

We have seen Stemming in detail in the Porter Stemmer Assignment & in this project we see how to extract tokens, remove punctuation and remove stopwords.

Stopwords are a set of words in a language that are used very frequently in that language and provide grammatical and syntactic structure to that particular language.

Although computationally speaking these words do not add a lot of meaning to a particular sentence and often when we wish to extract information from a particular document removing these stopwords can help us make the document more information dense and now we have a corpus from which information extraction and sentiment analysis can be performed with better results.

In this assignment I introduce the NLTK (Natural language Toolkit) Python Library which we will use boh for removing stopwords and removing punctuations from a sample text file; my Resume in this case.

Some Standard steps that are often followed in a Natural Language Processing pre-processing pipeline are:

1. Converting the corpus into lowercase
2. Extracting tokens (words from the corpus)
3. Removing punctuations from the selected tokens.
4. And using a stemming algorithm (like Porter Stemmer) as was implemented in the last assignment.

In this assignment the removal of punctuations, tokenization and removal of stopwords has been shown.

Input (Resume/CV) [See resume.txt on GitHub]

Anish Sachdeva

Software Developer + Clean Code Enthusiast

Phone : 8287428181

email : anish_@outlook.com

home : sandesh vihar, pitampura, new delhi - 110034

date of birth : 7th April 1998

languages : English, Hindi, French

Work Experience

What After College (4 months)

Delhi, India

Creating content to teach Core Java and Python with Data Structures and Algorithms and giving online classes to students

Summer Research Fellow at University of Auckland (2 Months)

Auckland, New Zealand

Worked on Geometry of Mobius Transformations, Differential Grometry under Dr. Pedram Hekmati at the Department of Mathematics, University of Auckland

Software Developer at CERN (14 Months)

CERN, Geneva, Switzerland

Worked in the core Platforms team of the FAP-BC group. Part of an agile team of developers that maintains and adds core functionality to applications used internally at CERN by HR, Financial, Administrative and other departments including Scientific

Worked on legacy applications that comprise of single and some times multiple frameworks such as Java Spring, Boot, Hibernate and Java EE. Also worked with Google Polymer 1.0 and JSP on the client side

Maintained CERN's Electronic Document Handing System application with >1M LOC that comprising of multiple frameworks and created ~20 years ago. Worked on feature requests, support requests and incidents and also release cycles

Teaching Assistant (4 Months)

Coding Ninjas, Delhi

Served as the teaching assistant to Nucleus - Java with DS batch, under Mr. Ankur Kumar. Worked on creating course content and quizzes for online platform of Coding Ninjas for Java. Helped students in core Data Structures and Algorithms concepts in Java

Education

Delhi Technological University (2016 - 2021)

Bachelors of Technology Mathematics and Computing

CGPA: 9.2

The Heritage School Rohini (2004 - 2016)

Physics, Chemistry, Maths + Computer Science with English

Senior Secondary: 94.8%

Secondary: 9.8 CGPA

Technical Skills

Java + Algorithms and Data Structures

MEAN Stack Web Development

Python + Machine Learning

MATLAB + Octave

MySQL, PostgresSQL & MongoDB

Other Skills

MS Office, Adobe Photoshop, LaTeX + MiTeX

University Courses

Applied Mathematics I, II, III

Linear Algebra + Probability & Statistics + Stochastic Processes + Discrete Maths

Computer Organization & Architecture + Data Structures + Algorithm Design and Analysis + DBMS + OS

Computer Vision + NLP

Important Links

https://www.linkedin.com/in/anishsachdeva1998/

https://github.com/anishLearnsToCode

https://www.hackerrank.com/anishviewer

Code

Utils.py is a file that has 2 methods. **`tokenize_document`** takes a document as a `string` and returns a list of all tokens after removing punctuations and stopwords. The other method **`tokenize_document_pretty`** takes a document as a `string` and returns the formatted version of the document also as a string with punctuations and stopwords removed.

utils.py [See on GithHub]

```
from nltk.corpus import stopwords

from nltk.tokenize import word_tokenize

en_stopwords = set(stopwords.words('english'))



def tokenize_document(document, tokenizer=word_tokenize):

  tokens = tokenizer.tokenize(document)

  return [token for token in tokens if token not in en_stopwords]



def tokenize_document_pretty(document, tokenizer=word_tokenize):

  result = []

  for line in document.split('\n'):

     result.append(' '.join(tokenize_document(line, tokenizer)))

  return '\n'.join(result)
```

`driver.py` is the main driver file of this program which will import the functions from utils and will also create a tokenizer using **`nltk.RegexpTokenizer`** factory method and will import the resume file and then call the **`tokenize_document`** method and save the tokenized list of the resume in a *pickle* file so that analytics can be run on it. It will also print all tokens.

driver.py *[See Code on GitHub]*

```
from porter_stemmer import PorterStemmer

import nltk
import pickle
from utils import tokenize_document

resume_file = open('../assets/resume.txt', 'r')
resume = resume_file.read()
resume_file.close()

tokenizer = nltk.RegexpTokenizer(r'\w+')
resume_tokenized = tokenize_document(resume, tokenizer)
print(resume_tokenized)
pickle.dump(resume_tokenized, open('../assets/resume_tokens.p', 'wb'))
```

output.py *[See Code on GitHub]*

The **`output.py`** file reads in the **`resume.txt`** file and uses the **`tokenize_document_pretty`** defined in the **`utils.py`** file to print a well formatted output with stop words and punctuations removed.

```
import nltk
from utils import tokenize_document_pretty

tokenizer = nltk.RegexpTokenizer(r'\w+')
resume_file = open('../assets/resume.txt', 'r')
resume = resume_file.read()
resume_file.close()

print(tokenize_document_pretty(resume, tokenizer))
```

analytics.py *[See Code on GitHub]*

The analytics file loads in the resume_tokens.p file that was created by the driver program and performs very basic operations on the tokens to see what are the frequencies of the most commonly occurring tokens and also to see the number of unique tokens.

```
import pickle
from utils import tokenize_document, tokenizer
from collections import Counter

tokens_with_stopwords = tokenize_document(open('../assets/resume.txt', 'r').read(), tokenizer,
remove_stopwords=False)
print('Number of Tokens: (with stopwords)', len(tokens_with_stopwords))
print('Number of Unique tokens: (with stopwords)', len(set(tokens_with_stopwords)), '\n')

tokens = pickle.load(open('../assets/resume_tokens.p', 'rb'))
print('Number of Tokens: (without stopwords)', len(tokens))
tokens_set = set(tokens)
print('Number of Unique tokens: (without stopwords)', len(tokens_set))

frequencies = Counter(tokens)
print('\nThe Frequencies of the most common 10 tokens are: (in tokens without stopwords)\n',
frequencies.most_common(10))
```

Output

```
C:\Users\anish\projects\nlp\stop-words-removal\venv\Scripts\python.exe
C:/Users/anish/projects/nlp/stop-words-removal/src/output.py

Anish Sachdeva

Software Developer Clean Code Enthusiast


Phone 8287428181

email anish_ outlook com

home sandesh vihar pitampura new delhi 110034

date birth 7th April 1998

languages English Hindi French


Work Experience

What After College 4 months

Delhi India

Creating content teach Core Java Python Data Structures Algorithms giving online classes
students


Summer Research Fellow University Auckland 2 Months

Auckland New Zealand

Worked Geometry Mobius Transformations Differential Grometry Dr Pedram Hekmati
Department Mathematics University Auckland


Software Developer CERN 14 Months

CERN Geneva Switzerland

Worked core Platforms team FAP BC group Part agile team developers maintains adds core
functionality applications used internally CERN HR Financial Administrative departments
including Scientific

Worked legacy applications comprise single times multiple frameworks Java Spring Boot
Hibernate Java EE Also worked Google Polymer 1 0 JSP client side
```

Maintained CERN Electronic Document Handing System application 1M LOC comprising multiple frameworks created 20 years ago Worked feature requests support requests incidents also release cycles

Teaching Assistant 4 Months

Coding Ninjas Delhi

Served teaching assistant Nucleus Java DS batch Mr Ankur Kumar Worked creating course content quizzes online platform Coding Ninjas Java Helped students core Data Structures Algorithms concepts Java

Education

Delhi Technological University 2016 2021

Bachelors Technology Mathematics Computing

CGPA 9 2

The Heritage School Rohini 2004 2016

Physics Chemistry Maths Computer Science English

Senior Secondary 94 8

Secondary 9 8 CGPA

Technical Skills

Java Algorithms Data Structures

MEAN Stack Web Development

Python Machine Learning

MATLAB Octave

MySQL PostgresSQL MongoDB

Other Skills

MS Office Adobe Photoshop LaTeX MiTeX

University Courses

Applied Mathematics I II III

Linear Algebra Probability Statistics Stochastic Processes Discrete Maths

Computer Organization Architecture Data Structures Algorithm Design Analysis DBMS OS

Computer Vision NLP

Important Links

https www linkedin com anishsachdeva1998

https github com anishLearnsToCode

https www hackerrank com anishviewer

Analytics

We run the analytics.py file and the get the following output given below:

```
Number of Tokens: (with stopwords) 363

Number of Unique tokens: (with stopwords) 246


Number of Tokens: (without stopwords) 293

Number of Unique tokens: (without stopwords) 228

Percentage Reduction in tokens after removing stopwords:
7.894736842105263


The Frequencies of the most common 10 tokens are: (in tokens without
stopwords)

 [('Java', 7), ('Worked', 5), ('com', 4), ('Data', 4), ('Structures',
4), ('University', 4), ('CERN', 4), ('Delhi', 3), ('Algorithms', 3),
('Auckland', 3)]
```

Using this output we can clearly see that the total number of tokens is considerably reduced after removing the stopwords. The total percentage reduction in stopwords is approximately ~7.89% and this number should be about the same for any standard resume. By removing the stop words we are effectively reducing the size of our training set and hence all other applications/processes that we run on this set after the pre-processing phase should run about ~8% faster.

Hence, removing these stopwords that do not add a lot of syntactical meaning to our data and do not provide extra information can be a beneficial step.

Discussion

We have seen that after removing the stopwords from the resume our number of words has gone down considerably and also that the words we removed never added a lot of meaning to our text. Large companies who will receive many resumes will want to search them using keywords such as Java, Python, web Development etc. and words such as i, me, mine are superfluous in nature.

So, by removing these stopwords we have actually made our corpus more information dense and any other further task we might perform such as converting these words into embeddings or any other Machine Learning/Deep Learning task will now be done on a smaller Corpus and hence would run faster.

After removing the stop words we can also run frequency analysis on our data-set better. For example when we now run a frequency analysis to see what was the most common listed word (or the top 5 occurring words) in any resume the resus we receive will most likely be the top skills and top experiences that the candidate wished to mention.

After running a frequency analysis on my resume the top 10 things include the terms Java, Python, CERN, Auckland, Data Structures and Algorithms and these are indeed the top skills that I possess and I would like prospective employers to know. We also see that CERN and Auckland have popped up. I was the only student selected to work at CERN from India in the year 2018 and yes, this is something I would like to highlight in my resume. (same case for the Summer Research Program at Auckland).

So, by using stopwords and eliminating extraneous terms we can highlight the top skills and experiences in any candidate's resume. This has alst revealed some flaws in the list of stopwords provided by the nltk library. That list doesn't include the word 'Worked' and hence that has shown up as one of my top frequency words in my resume.

We need to update the list of stopwords for this specific domain of resume analysis. Weare very certain when someone lists something in their resume they most definitely have worked there and they most definitely have some job/internship experience and only then have they listed that in their resume. The other scenario would simply mean lying.

So we have to update the list of stopwords for our task (and even update them in any other domain we process data for). The updated list of stopwords can include 'work', 'worked', 'job', 'internship' etc. because if someone lists a job 10 times on their resume we can't infer a lot from that but seeing docker and Python and System Analysis on the top will definitely mean something.

Overall stopwords has allowed us to retrieve information just by using simple things as frequency analysis and further applications such as extracting word embeddings and information extraction will also benefit from the removal of noisy data and hence considering these above advantages removing stopwords is a very beneficial pre-processing step.