

Natural Language Processing with SpaCy

What even is it?

Natural Language Processing (NLP) is a tool similar to regex, but allows us to take our text analytics to entirely new levels

What even is it?

| Regex | NLP |
|----------------------------------|---|
| Create patterns to match in text | Identify the structure of text and use that to refine information |
| Used to verify or find data | Used to analyze data |
| Applies user-defined rules | Relies heavily on ML-based (or other) models |

Tasks we can complete with NLP

- Identify parts of speech
- Iterate over sentences
- Find words used to describe various nouns (or anything else!)
- Visualize semantic relationships
- Filter text for analysis
- Conduct sentiment analysis

Getting Started

First, we will need to install the right libraries (and a corpus!):

```
# Load the SpaCy library (IF NEEDED)
!pip install spacy

# Load the corpus
!python -m spacy download en_core_web_sm
```

A corpus is the body of knowledge that our NLP library (`spacy`) needs in order to be able to process the english language. We will use the smallest english corpus for now.

The `web` part means that it was trained on web content, and will perform best when applied to similar material.

Get some text!

```
import requests

jane = requests.get("https://github.com/dustywhite7/Econ8320/  
raw/master/AssignmentData/janeEyreCh1to3.txt").text
```

This file contains chapters 1 to 3 of Jane Eyre

(Lots of fun texts to play with at [Project Gutenberg](#))

Rev the NLP engines!

```
import spacy  
  
nlp = spacy.load("en_core_web_sm")  
doc = nlp(jane)
```

Here, we are pointing to the language file we want to use, and then parsing our text

Important attributes

Our new parsed document (`doc` in this case), has some important **attributes**:

- `sents` - a generator function to iterate over each sentence in the document
- `tokens` - each individual element of the document
 - Elements exist at the word/punctuation level
 - Access via `for i in doc` -style loops

[Some] Attributes of tokens

- `lemma_` - the "root word" from which a token/word is derived
- `pos_` - the part of speech of a token/word
- `dep_` - the relationship of dependent tokens to the parent token (adjectives to nouns, etc.)
- `like_email` / `like_num` / `like_url` - check if a token is like an email, number, or url (unlikely in Jane Eyre)

Doing NLP

```
[i.lemma_ for i in doc if (not i.is_punct) and (not i.is_space)][:100]
```

```
['chapter',  
 'I',  
 'there',  
 'be',  
 'no',  
 'possibility',  
 'of',  
 'take',  
 'a',  
 'walk',  
 'that',  
 ...]
```

What is happening here??

Lemmatization

When we have words with many forms (verbs, plurals, etc.), we want to work with the **lemma** of that word, so that we can recognize that all forms are actually the same word being used in different ways.

"Ran" --> "run"

"Are" --> "be"

Noun chunks

Sometimes, you want to be able to see a "complete" noun, and noun chunks are the tool to use!

```
[i for i in doc.noun_chunks]
```

Noun chunks include all of the modifiers for a given noun, and make it easier to build a more complete understanding of the references being made.

Mapping a sentence

```
from spacy import displacy  
  
sent = [i for i in doc.sents][100]  
displacy.serve(sent, style="dep")
```

Sentiment Analysis

Here, we are going to use `spacytextblob` to increase the functionality of our NLP model. Install with the following command:

```
! pip install pip install spacytextblob
```

Note: only use the `!` characters if installing from inside of jupyter notebook or other python interpreter (not when installing from shell/command prompt)

Use `spacytextblob` for sentiment analysis

```
import spacy
from spacytextblob.spacytextblob import SpacyTextBlob
import requests

jane = requests.get(
    "https://github.com/dustywhite7/Econ8320/raw/master/AssignmentData/janeEyreCh1to3.txt").text

nlp = spacy.load('en_core_web_sm')
nlp.add_pipe('spacytextblob')
blob = nlp(jane)

for sentence in blob.sents:
    print("Polarity: {0:3.2f},
          Subjectivity: {1:3.2f}".format(
            sentence._.polarity,
            sentence._.subjectivity))
```

Lab Time!