Assignment_102 Reflection

Intro to NLP & LLMs - Fall 2025

"*Which library provided more consistent results for your text, and why might results vary across domains?*"

For my text, I thought that spaCy was more correct most of the time. By just picking the first 3 words "Hateful", "creator/creator!", "Accursed/'Accursed", 2 of the POS tags marked by spaCy were more correct than the POS tags assigned by the NLTK. It surprised me that capitalization and punctuation would have such a big impact on the tagging of what are otherwise 2 identical words. This is likely the reason why the assignment recommended beforehand that all the text be lowercased and the punctuations stripped. It really makes a difference!

As explained in the practice jupyter notebook "102_text_preprocessing.ipynb", NLTK is the traditional learning tool that has each possible step of natural language processing such as tokenization, lemmatization, POS-tagging all separated out. I think in a NLP academic/research domain, this is where NLTK will be more helpful because then you can study and learn about how each step of the process (tokenization, lemmatization, POS-tagging) individually affects the final output.

Nevertheless, in the industry, I think spaCy would be more helpful because just writing the 2 lines below:

```
spacy_model = spacy.load('some_model_name')
document = spacy_model(text)
```

The model automatically handles much of the tasks for you such as returning a document type object containing each token which in it of itself contains the word's assigned POS tag, its lemma, whether it is considered a stopword or not, etc. This makes it much more efficient to extract basic token information to aid in any kind of NLP decision-making. A potential issue that could arise from this is that one can easily overlook the aggressiveness of the preprocessing because it is so easy to just instinctively run them. For example, spaCy considered "make" as a stopword, even though in the context of my provided text, the word "make" has its purpose ("*... God, in pity,* ***made*** *man beautiful and alluring, after his own image;...*") without the word make/made, it is more difficult to understand the meaning behind the sentence: "... God, in pity, man beautiful and alluring...". This could also be applied to the medical domain for example if a text was "Patient A made their own medicine." then aggressive spaCy preprocessing may convert it to "Patient A their own medicine." and without that "made" word, it is easy to assume the original text was simply "Patient A had their own medicine" which informs medical workers of 2 very distinct pieces of knowledge about their patient.