

Allen Hoskins

DS7337: Natural Language Processing, Fall 2022

Homework 4

```
In [ ]: from nltk import tag, word_tokenize, pos_tag
import spacy
from spacy.tokenizer import Tokenizer
sp = spacy.load('en_core_web_sm')
```

Part 1: Run one of the part-of-speech (POS) taggers available in Python.

- Find the longest sentence you can, longer than 10 words, that the POS tagger tags correctly. Show the input and output.
- Find the shortest sentence you can, shorter than 10 words, that the POS tagger fails to tag 100 percent correctly. Show the input and output. Explain your conjecture as to why the tagger might have been less than perfect with this sentence.

The short sentence that was used was "The robber eats, shoots and leaves." This is a play on the commonly used sentence "The panda eats, shoots and leaves," which is used to describe the importance of Oxford commas. Since shoots and leaves can be both nouns and verbs, the tagger has difficulty correctly tagging the sentence. It currently tags `shoots` and `leaves` as plural nouns, but in the context of the sentence, they should be verbs.

```
In [ ]: long_sentence = 'Their plots were failing because of some important friends of
```

```
In [ ]: ls_tagged = pos_tag(word_tokenize(long_sentence))
print('Long Sentence NLTK Tagging:\n')
ls_tagged
```

Long Sentence NLTK Tagging:

```
Out[ ]: [('Their', 'PRP$'),
        ('plots', 'NNS'),
        ('were', 'VBD'),
        ('failing', 'VBG'),
        ('because', 'IN'),
        ('of', 'IN'),
        ('some', 'DT'),
        ('important', 'JJ'),
        ('friends', 'NNS'),
        ('of', 'IN'),
        ('the', 'DT'),
        ('king', 'NN'),
        ('.', '.')]

```

```
In [ ]: short_sentence = 'The robber eats, shoots and leaves.'
```

```
In [ ]: ss_tagged = pos_tag(word_tokenize(short_sentence))
print('Short Sentence NLTK Tagging:\n')
ss_tagged
```

Short Sentence NLTK Tagging:

```
Out[ ]: [('The', 'DT'),
        ('robber', 'NN'),
        ('eats', 'NNS'),
        (',', ','),
        ('shoots', 'NNS'),
        ('and', 'CC'),
        ('leaves', 'NNS'),
        ('.', '.')]

```

Part 2: Run a different POS tagger in Python.

Process the same two sentences from question 1.

- a. Does it produce the same or different output?

When running the tagger from the Python package `SpaCy`, the long sentence is tagged the same way as `nltk` tagged it, however in the short sentence, the word `eats` is correctly tagged when using `SpaCy` but not when utilizing `nltk`.

- b. Explain any differences as best you can.

`nltk` incorrectly tagged the word `eats` as a plural noun and this could be due to assuming it was used in the same way as `food`, but due to the context of this sentence it is expected to struggle to correctly tag `shoots` and `leaves` as verbs.

```
In [ ]: spacy_long = sp('Their plots were failing because of some important friends of
print('Long Sentence SpaCy tagging:\n')
for word in spacy_long:
    print(f'{word.text:{12}} {word.pos_:{10}} {word.tag_:{8}} {spacy.explain(wc

```

Long Sentence SpaCy tagging:

Their	PRON	PRP\$	pronoun, possessive
plots	NOUN	NNS	noun, plural
were	AUX	VBD	verb, past tense
failing	VERB	VBG	verb, gerund or present participle
because	SCONJ	IN	conjunction, subordinating or preposition
of	ADP	IN	conjunction, subordinating or preposition
some	DET	DT	determiner
important	ADJ	JJ	adjective (English), other noun-modifier (Chinese)
friends	NOUN	NNS	noun, plural
of	ADP	IN	conjunction, subordinating or preposition
the	DET	DT	determiner
king	NOUN	NN	noun, singular or mass
.	PUNCT	.	punctuation mark, sentence closer

```
In [ ]: spacy_short = sp('The panda eats, shoots and leaves.')
print('Short Sentence SpaCy tagging:\n')
for word in spacy_short:
    print(f'{word.text:{12}} {word.pos_:{10}} {word.tag_:{8}} {spacy.explain(word)}
```

Short Sentence SpaCy tagging:

The	DET	DT	determiner
panda	NOUN	NN	noun, singular or mass
eats	VERB	VBZ	verb, 3rd person singular present
,	PUNCT	,	punctuation mark, comma
shoots	NOUN	NNS	noun, plural
and	CCONJ	CC	conjunction, coordinating
leaves	NOUN	NNS	noun, plural
.	PUNCT	.	punctuation mark, sentence closer

Part 3: In a news article from this week's news, find a random sentence of at least 10 words.

- Looking at the Penn tag set, manually POS tag the sentence yourself.

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

- Now run the same sentences through both taggers that you implemented for questions

- Explain any differences between the two taggers and your manual tagging as much as you can.

The primary difference between my own tagging utilizing the Penn tag set, the nltk tagging, and SpaCy tagging is found in the word `mainland`. In the context of the sentence, `mainland` is an adjective that describes a location in China. The `SpaCy` tagger flags `mainland` as a noun. While all other words are correctly tagged, this word can be both an adjective or noun, and any tagger might struggle with this word.

```
In [ ]: news = 'Google discontinued its Google Translate service in mainland China citi
```

```
In [ ]: own_tag = [ ("Google", 'NNP'),
                    ("discontinued", 'VBD'),
                    ("its", 'PRP$'),
                    ("Google", 'NNP'),
                    ("Translate", 'NNP'),
                    ("service", 'NN'),
                    ("in", 'IN'),
                    ("mainland", 'JJ'),
                    ("China", 'NNP'),
                    ("citing", 'VBG'),
                    ("low", 'JJ'),
                    ("usage", 'NN'),
                    (".", 'SYM')]

print('Own tagging from Penn tag set: \n')
own_tag
```

Own tagging from Penn tag set:

```
Out[ ]: [('Google', 'NNP'),
          ('discontinued', 'VBD'),
          ('its', 'PRP$'),
          ('Google', 'NNP'),
          ('Translate', 'NNP'),
          ('service', 'NN'),
          ('in', 'IN'),
          ('mainland', 'JJ'),
          ('China', 'NNP'),
          ('citing', 'VBG'),
          ('low', 'JJ'),
          ('usage', 'NN'),
          ('.', 'SYM')]
```

```
In [ ]: news_tagged = pos_tag(word_tokenize(news))
print('NLTK tagging: \n')
news_tagged
```

NLTK tagging:

```
Out[ ]: [('Google', 'NNP'),
          ('discontinued', 'VBD'),
          ('its', 'PRP$'),
          ('Google', 'NNP'),
          ('Translate', 'NNP'),
          ('service', 'NN'),
          ('in', 'IN'),
          ('mainland', 'JJ'),
          ('China', 'NNP'),
          ('citing', 'VBG'),
          ('low', 'JJ'),
          ('usage', 'NN'),
          ('.', '.')]


```

```
In [ ]: spacy_news = sp('Google discontinued its Google Translate service in mainland C
print('SpaCy Tagging:\n')
for word in spacy_news:
    print(f'{word.text:{12}} {word.pos_{10}} {word.tag_{8}} {spacy.explain(wc
```

SpaCy Tagging:

Google	PROPN	NNP	noun, proper singular
discontinued	VERB	VBD	verb, past tense
its	PRON	PRP\$	pronoun, possessive
Google	PROPN	NNP	noun, proper singular
Translate	PROPN	NNP	noun, proper singular
service	NOUN	NN	noun, singular or mass
in	ADP	IN	conjunction, subordinating or preposition
mainland	NOUN	NN	noun, singular or mass
China	PROPN	NNP	noun, proper singular
citing	VERB	VBG	verb, gerund or present participle
low	ADJ	JJ	adjective (English), other noun-modifier (Chi
nese)			
usage	NOUN	NN	noun, singular or mass
.	PUNCT	.	punctuation mark, sentence closer