#SpaCY

SpaCy is probably the main competitor to NLTK. It is faster in most cases, but it only has a single implementation for each NLP component. Also, it represents everything as an object rather than a string, which simplifies the interface for building applications.

This also helps it integrate with many other frameworks and data science tools, so you can do more once you have a better understanding of your text data.

However, SpaCy doesn't support as many languages as NLTK.

It does have a simple interface with a simplified set of choices and great documentation, as well as multiple neural models for various components of language processing and analysis.

Overall, this is a great tool for new applications that need to be performant in production and don't require a specific algorithm.

#SpaCY vs NLTK

--->SpaCY spaCy takes an object oriented approach.

--->NLTK NLTK is essentially a string processing library, where each function takes strings as input and returns a processed string.

In [1]:

```
'''Tokenization is always the first step before we can do any text data processing.
What this means is that spaCy will segment sentences into words, punctuations,
symbols and others by applying specific rules to each language.'''
#python -m spacy download en_core_web_sm
import spacy
nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
for token in doc:
    print(token.text)
```

```
Apple
is
looking
at
buying
U.K.
startup
for
$
1
billion
```

In [2]:

```
'''Once we have tokenized words,
the typical next step would be to perform Part-of-speech (POS)
tagging to understand the grammatical properties (noun, verb, adjective etc.) of each word.
import spacy
nlp= spacy.load("en_core_web_sm")
doc= nlp("She ate the pizza")
for token in doc:
    print(token.text, token.pos_)
```

```
She PRON
ate VERB
the DET
pizza NOUN
```

In [3]:

```
'''Named Entity Recognition
Named entities are “real world objects” that are assigned a name — for example, a person, a
import spacy
nlp= spacy.load("en_core_web_sm")
doc= nlp(u"Apple is looking at buying U.K. startup for $1 billion")
for ent in doc.ents:
    print(ent.text, ent.label_)
```

```
Apple ORG
U.K. GPE
$1 billion MONEY
```

In [4]:

```
'''Similarity By default, the similarity returned by spaCy is the cosine similarity between
import spacy
# Load a larger model with vectors
nlp = spacy.load('en_core_web_md')
# Compare two documents
doc1 = nlp("I love junk food")
doc2 = nlp("I ride bikes")
print(doc1.similarity(doc2))
```

```
0.625923210847021
```

Refernces

1. https://towardsdatascience.com/so-whats-spacy-ad65aa1949e0 (https://towardsdatascience.com/so-whats-spacy-ad65aa1949e0)
2. https://spacy.io/usage/examples (https://spacy.io/usage/examples)
3. https://opensource.com/article/19/3/natural-language-processing-tools (https://opensource.com/article/19/3/natural-language-processing-tools)
4. https://github.com/explosion/spaCy/issues/3923 (https://github.com/explosion/spaCy/issues/3923)