Tokenizing

In [ ]:
```
# import pip
# pip.main(["install","nltk"])
```

In [ ]:
```
from nltk.tokenize import sent_tokenize, word_tokenize
```

In [ ]:
```
sample = "Muad'Dib learned rapidly because his first training was in how to le
arn. And the first lesson of all was the basic trust that he could learn. It's
shocking to find how many people do not believe they can learn,and how many mo
re believe learning to be difficult."
```

In [ ]:
```
print(sample)
```

In [ ]:
```
sent_tokenize(sample)
```

In [ ]:
```
word_tokenize(sample)
```

Stop Words

In [ ]:
```
import nltk
nltk.download("stopwords")
```

In [ ]:
```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

In [ ]:
```
sample="Sir, I protest. I am not a merry man!"
words = word_tokenize(sample)
words
```

In [ ]:
```
stop_words = set(stopwords.words("english"))
```

In [ ]:
```
print(stop_words)
```

In [ ]:
```
filtered_list = []
```

In [ ]:
```
for word in words:
    if word.casefold() not in stop_words:
        filtered_list.append(word)
```

In [ ]:
```
filtered_list
```

Stemming

In [ ]:
```
from nltk.stem import PorterStemmer,SnowballStemmer
```

In [ ]:
```
stemmer = PorterStemmer()
```

In [ ]:
```
sample = "The crew of the USS Discovery discovered many discoveries. Discoveri
ng is what explorers do."
```

In [ ]:
```
words = word_tokenize(sample)
words
```

In [ ]:
```
stemmed_words = [stemmer.stem(word) for word in words]
stemmed_words
```

In [ ]:
```
stemmer2 = SnowballStemmer("english")
```

In [ ]:
```
stemmed_words = [stemmer2.stem(word) for word in words]
stemmed_words
```

POS Tagging

In [ ]:
```
sample = "If you wish to make an apple pie from scratch, you must first invent
the universe."
```

In [ ]:
```
from nltk import pos_tag
```

In [ ]:
```
words = word_tokenize(sample)
words
```

In [ ]:
```
pos_tag(words)
```

In [ ]:
```
nltk.help.upenn_tagset()
```

Lemmatizing

In [ ]:
```
from nltk.stem import WordNetLemmatizer
```

In [ ]:
```
lemmatizer = WordNetLemmatizer()
```

In [ ]:
```
sample = "The friends of DeSoto love scarves."
```

In [ ]:
```
words = word_tokenize(sample)
```

In [ ]:
```
lemmatized_words = [lemmatizer.lemmatize(word) for word in words]
lemmatized_words
```

Chunking

In [ ]:
```
sample = "It's a dangerous business, Frodo, going out your door."
```

In [ ]:
```
words = word_tokenize(sample)
words
```

In [ ]:
```
tagpos=nltk.pos_tag(words)
tagpos
```

In [ ]:
```
grammar = "NP: {<DT>?<JJ>*<NN>}"
#NP stands for noun phrase
#Start with an optional (?) determiner ('DT')
#Can have any number (*) of adjectives (JJ)
#End with a noun (<NN>)
```

In [ ]:
```
chunk_parser = nltk.RegexpParser(grammar)
```

In [ ]:
```
tree = chunk_parser.parse(tagpos)
```

In [ ]:
```
tree.draw()
```

Chinking

In [ ]:
```
grammar = """
Chunk: {<.*>+}
}<JJ>{"""
```

In [ ]:
```
chunk_parser = nltk.RegexpParser(grammar)
```

In [ ]:
```
tree = chunk_parser.parse(tagpos)
```

In [ ]:
```
tree.draw()
```

Named Entity Recognition (NER)

In [ ]:
```
tree = nltk.ne_chunk(tagpos)
```

In [ ]:
```
tree.draw()
```

In [ ]:
```
msg="Obama was the president of America."
```

In [ ]:
```
words = word_tokenize(msg)
words
```

In [ ]:
```
tagpos=nltk.pos_tag(words)
tagpos
```

In [ ]:
```
tree = nltk.ne_chunk(tagpos)
```

In [ ]:
```
tree.draw()
```