```
import spacy

# Load the spaCy English language model
nlp = spacy.load("en_core_web_sm")

# Example document (replace with your own)
text = """Prashant Sarvaiya is a developer.
He loves building applications.
He works on IoT projects.
He develops apps using Flutter.
Prashant explores AI and ML.
He studies at Marwadi University.
He enjoys solving real problems.
He participates in hackathons.
He learns cloud technologies.
Prashant aims to create impact."""


# Create a spaCy Doc object
doc = nlp(text)

# Print the tokens
for token in doc:
    print(token.text)
```

```
He
works
on
IoT
projects
.


He
develops
apps
using
Flutter
.


Prashant
explores
AI
and
ML
.


He
studies
at
Marwadi
University
.


He
enjoys
solving
real
problems
.
```

```
Prashant
aims
to
```

```
# Remove specific characters or words
filtered_text = text.replace(".", "")  # Example: removing periods
filtered_words = [word for word in doc if word.text != "is"]  # Example: removing the word "is"
filtered_words
```

```
,
He,
loves,
building,
applications,
.,
,
He,
works,
on,
IoT,
projects,
.,
,
He,
develops,
apps,
using,
Flutter,
.,
,
Prashant,
explores,
AI,
and,
ML,
.,
,
He,
studies,
at,
Marwadi,
University,
.,
,
He,
enjoys,
solving,
real,
problems,
.,
,
He,
participates,
in,
hackathons,
.,
,
He,
learns,
cloud,
technologies,
.,
,
Prashant,
aims,
to,
create,
impact,
```

```
from spacy.lang.en.stop_words import STOP_WORDS

# Remove stop words
filtered_words = [word for word in doc if not word.is_stop]

# Print the filtered words
for word in filtered_words:
    print(word.text)
```

```
Sarvaiya
developer
.
```

```
loves
building
applications
.


works
IoT
projects
.


develops
apps
Flutter
.


Prashant
explores
AI
ML
.


studies
Marwadi
University
.


enjoys
solving
real
problems
.


participates
hackathons
.


learns
cloud
technologies
.


Prashant
aims
create
impact
```

```
for token in doc:
    print(token.text, token.pos_)
```

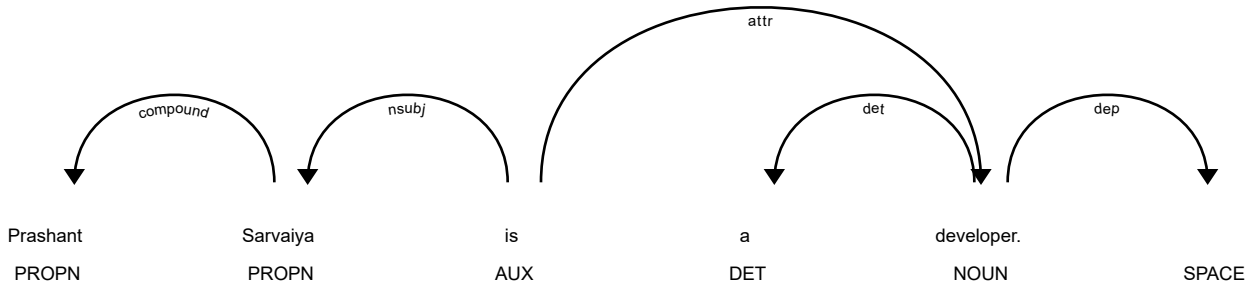
```
 SPACE
He PRON
works VERB
on ADP
IoT ADJ
projects NOUN
. PUNCT

 SPACE
He PRON
develops VERB
apps NOUN
```

```
 studies VERB
 at ADP
 Marwadi PROPN
 University PROPN
 . PUNCT

  SPACE
 He PRON
 enjoys VERB
 solving VERB
 real ADJ
 problems NOUN
 . PUNCT

  SPACE
 He PRON
 participates VERB
 in ADP
 hackathons NOUN
 . PUNCT

  SPACE
 He PRON
 learns VERB
 cloud ADJ
 technologies NOUN
 . PUNCT

  SPACE
 Prashant PROPN
 aims VERB
 to PART
```

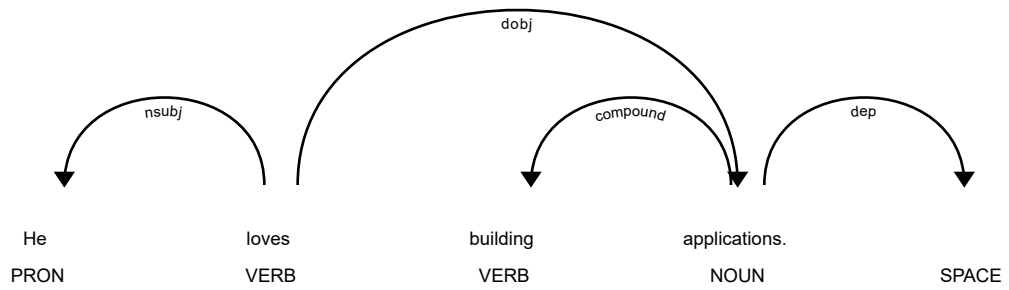
```
for chunk in doc.noun_chunks:
    print(chunk.text)
```

```
Prashant Sarvaiya
a developer
He
building applications
He
IoT projects
He
apps
Flutter
Prashant
AI
ML
He
Marwadi University
He
real problems
He
hackathons
He
cloud technologies
Prashant
impact
```

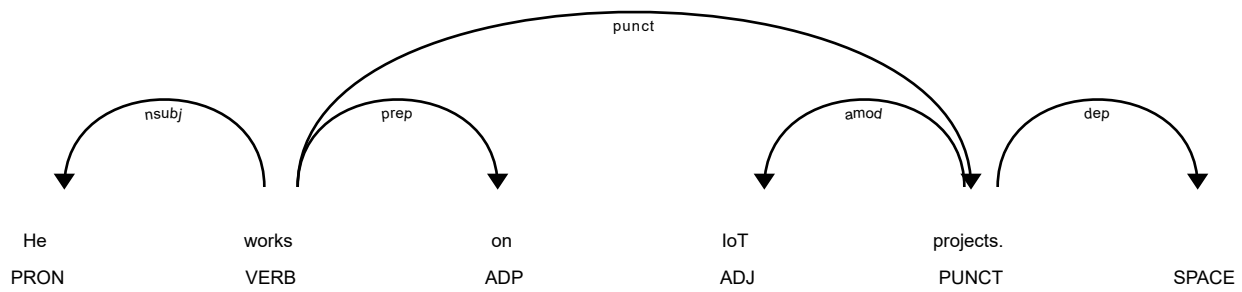
```
# Render each sentence separately
for sent in doc.sents:
    displacy.render(sent, style="dep", jupyter=True)
    print("\n" + "="*50 + "\n")  # Separator for better visualization
```


attr
compound
nsubj
det
dep
Prashant
PROPN
Sarvaiya
PROPN
is
AUX
a
DET
developer.
NOUN
SPACE

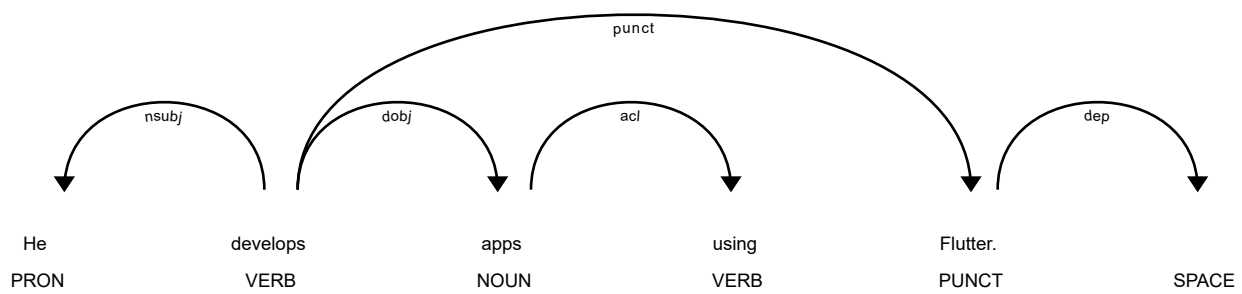
==========


dobj
nsubj
compound
dep
He
PRON
loves
VERB
building
VERB
applications.
NOUN
SPACE

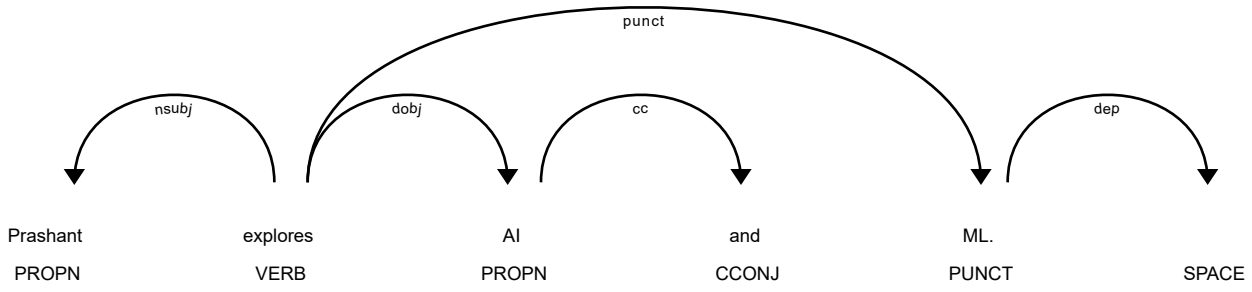
==========


punct
nsubj
prep
amod
dep
He
PRON
works
VERB
on
ADP
IoT
ADJ
projects.
PUNCT
SPACE

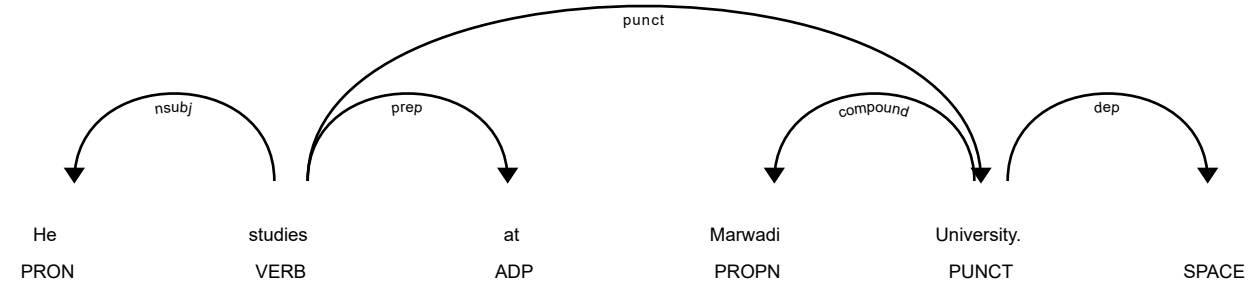
==========


punct
nsubj
dobj
acl
dep
He
PRON
develops
VERB
apps
NOUN
using
VERB
Flutter.
PUNCT
SPACE

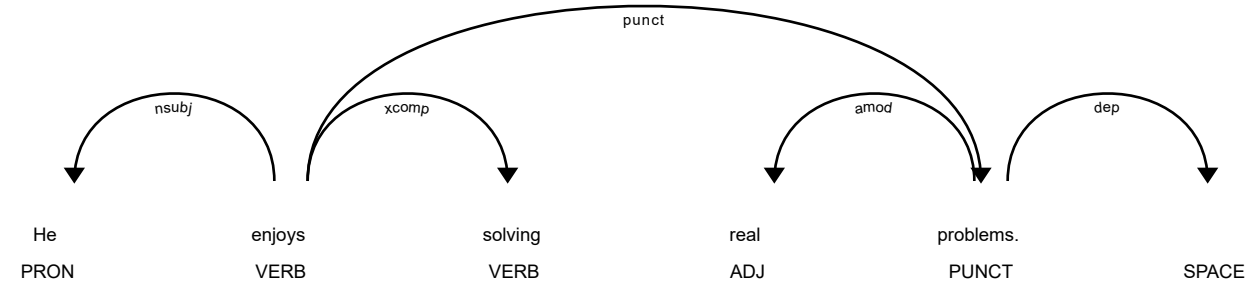
==========


punct
nsubj
dobj
cc
dep
Prashant
PROPN
explores
VERB
AI
PROPN
and
CCONJ
ML.
PUNCT
SPACE

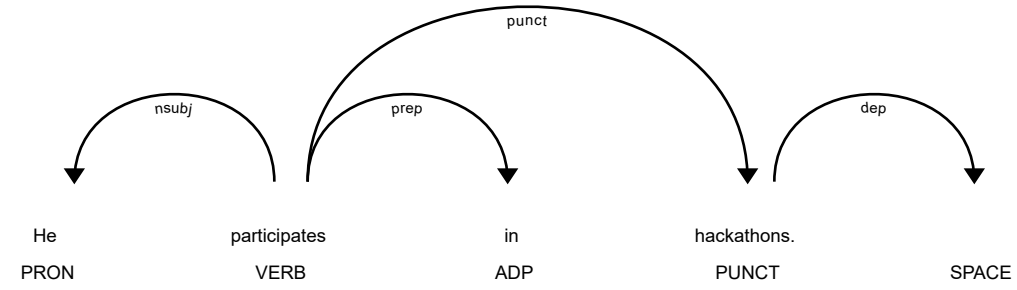
==========


punct
nsubj
prep
compound
dep
He
PRON
studies
VERB
at
ADP
Marwadi
PROPN
University.
PUNCT
SPACE

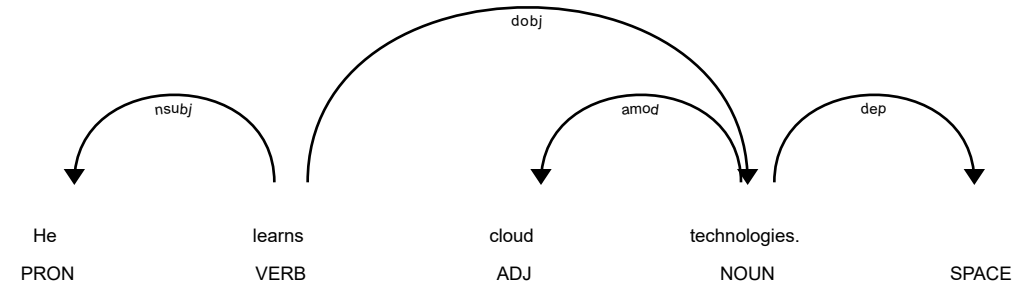
==========


punct
nsubj
xcomp
amod
dep
He
PRON
enjoys
VERB
solving
VERB
real
ADJ
problems.
PUNCT
SPACE

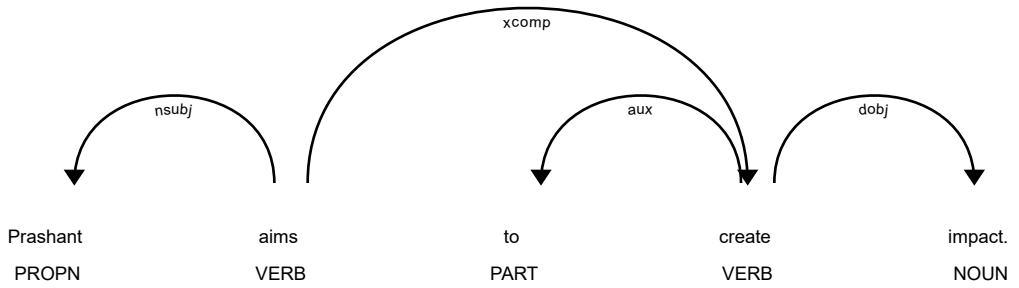
==========


punct
nsubj
prep
dep
He
PRON
participates
VERB
in
ADP
hackathons.
PUNCT
SPACE

==========




```
==================================================
```




```
==================================================
```

Start coding or generate with AI.

Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Artificial Intelligence (01CT0616)	**Aim:** To study Preprocessing of text (Tokenization. Filtration, Script Validation, Stop Word Removal, Stemming)	
Experiment No: 6	**Date:**	**Enrolment No: 92200133003**

Aim: To study Preprocessing of text (Tokenization. Filtration, Script Validation, Stop Word Removal, Stemming)

IDE: Google Colab

Theory:

To preprocess your text simply means to bring your text into a form that is predictable and analyzable for your task. A task here is a combination of approach and domain. Machine Learning needs data in the numeric form. We basically used encoding technique (Bag Of Word, Bi-gram, n-gram, TF-IDF, Word2Vec) to encode text into numeric vector. But before encoding we first need to clean the text data and this process to prepare (or clean) text data before encoding is called text preprocessing, this is the very first step to solve the NLP problems.

Tokenization:
Tokenization is about splitting strings of text into smaller pieces, or "tokens". Paragraphs can be tokenized into sentences and sentences can be tokenized into words.

Filtration:
Similarly, if we are doing simple word counts, or trying to visualize our text with a word cloud, stopwords are sonic of the most frequently occurring words but don't really tell us anything. We're often better off tossing the stopwords out of the text. By checking the Filter Stopwords option in the Text Pre-processing tool, you can automatically filter these words out.

Stemming:
Stemming is the process of reducing inflection in words (e.g. troubled, troubles) to their root form (e.g. trouble). The "root" in this case may not be a real root word, but just a canonical form of the original word.

Stemming uses a crude heuristic process that chops off the ends of words in the hope of correctly
actually be converted to troublinstead of trouble because the ends were just chopped off (ughh, how crude!). There are different algorithms for stemming. The most common algorithm, which is also known to be empirically effective for English, is Porters Algorithm. Here is an example of stemming in action with Porter Stemmer:

	original word	stemmed words
0	connect	connect
'I	connected	connect
2	connection	connect
3	connections	connect
4	connects	connect

Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Artificial Intelligence (01CT0616)	**Aim:** To study Preprocessing of text (Tokenization. Filtration, Script Validation, Stop Word Removal, Stemming)	
Experiment No: 6	**Date:**	**Enrolment No: 92200133003**

Stopword Removal

Stop words are a set of commonly used words in a language. Examples of stop words in English are "a", ‘‘the", "is", "are" and etc. The intuition behind using stop words is that, by removing low information words from text, we can focus on the important words instead.

For example, in the context of a search system, if your search query is 'what is text preprocessing?", you want the search system to focus on surfacing documents that talk about text preprocessing over documents that talk about what is. This can be done by preventing all words from your stop word list from being analyzed. Stop words are commonly applied in search systems, text classification applications, topic modeling, topic extraction and others. Stop word removal, while effective in search and topic extraction systems, showed to be non-critical in classification systems. However, it does help reduce the number of features in consideration which helps keep your models decently sized

Program (Code):

To be attached with

1. Tokenization
2. Filteration
3. StopWords Removal
4. PoS Tagging
5. Noun Phrase Chunking
6. Dependancy Parsing

Results:

To be attached with

Observation:

Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: Artificial Intelligence (01CT0616)	**Aim:** To study Preprocessing of text (Tokenization. Filtration, Script Validation, Stop Word Removal, Stemming)	
Experiment No: 6	**Date:**	**Enrolment No: 92200133003**

Post Lab Exercise:

1. Take your own document of 10 sentences and perform the same tasks. Attach code and screenshot of your output.
2. Write your observation for stemming and lemmatization you obtained for the sentences.

_____

_____

_____

_____

Basics of Natural LAnguage Processing

```
text = "I am learning AI Subject with the module of Natural Language Processing"
text
```

```
'I am learning AI Subject with the module of Natural Language Processing'
```

```
text.split()
```

```
['I',
 'am',
 'learning',
 'AI',
 'Subject',
 'with',
 'the',
 'module',
 'of',
 'Natural',
 'Language',
 'Processing']
```

```
'NaturalP' in text
```

```
False
```

```
words = text.split()
```

```
word = [w.lower() for w in words]
print(word)
```

```
['i', 'am', 'learning', 'ai', 'subject', 'with', 'the', 'module', 'of', 'natural', 'language', 'processing']
```

```
" ".join(word)
```

```
'i am learning ai subject with the module of natural language processing'
```

```
word = [w.upper() for w in words]
print(word)
```

```
['I', 'AM', 'LEARNING', 'AI', 'SUBJECT', 'WITH', 'THE', 'MODULE', 'OF', 'NATURAL', 'LANGUAGE', 'PROCESSING']
```

```
" ".join(word)
```

```
'I AM LEARNING AI SUBJECT WITH THE MODULE OF NATURAL LANGUAGE PROCESSING'
```

Punctuation Marks : !@#$%%^&*():{}><>

```
import string
string.punctuation
```

```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

```
text[5]
```

```
'l'
```

```
type(text)
```

```
str
```

```
import spacy
```

```
nlp = spacy.load('en_core_web_sm')
```

```
doc = nlp(text)
```

```
type(doc)
```

```
spacy.tokens.doc.Doc
```

```
text1 = "My name is Mukesh, learning Artificial Intelligence in MA112"
```

NER : Named Entity Recognition

```
doc = nlp(text1)
```

```
for token in doc.ents:
    print(token.text, token.label_)
```

```
Mukesh PERSON
Artificial Intelligence ORG
MA112 PRODUCT
Marwadi University ORG
Rajkot GPE
India GPE
11:00AM CARDINAL
```

```
doc = nlp("Red cars do not carry heigher insurance rate")
for chunk in doc.noun_chunks:
    print(chunk.text)
```

```
Red cars
heigher insurance rate
```

POS Tags : Part of speech tags

```
import nltk
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\prash\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
True
```

```
doc = nlp(text1)
```

```
for token in doc:
    print(token.text, ":" , token.pos_)
```

```
My : PRON
name : NOUN
is : AUX
Mukesh : PROPN
, : PUNCT
learning : VERB
Artificial : PROPN
Intelligence : PROPN
in : ADP
MA112 : PROPN
of : ADP
Marwadi : PROPN
University : PROPN
, : PUNCT
Rajkot : PROPN
, : PUNCT
India : PROPN
from : ADP
11:00AM : PROPN
```

```
#stopWords
```

Start coding or generate with AI.

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\prash\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
True
```

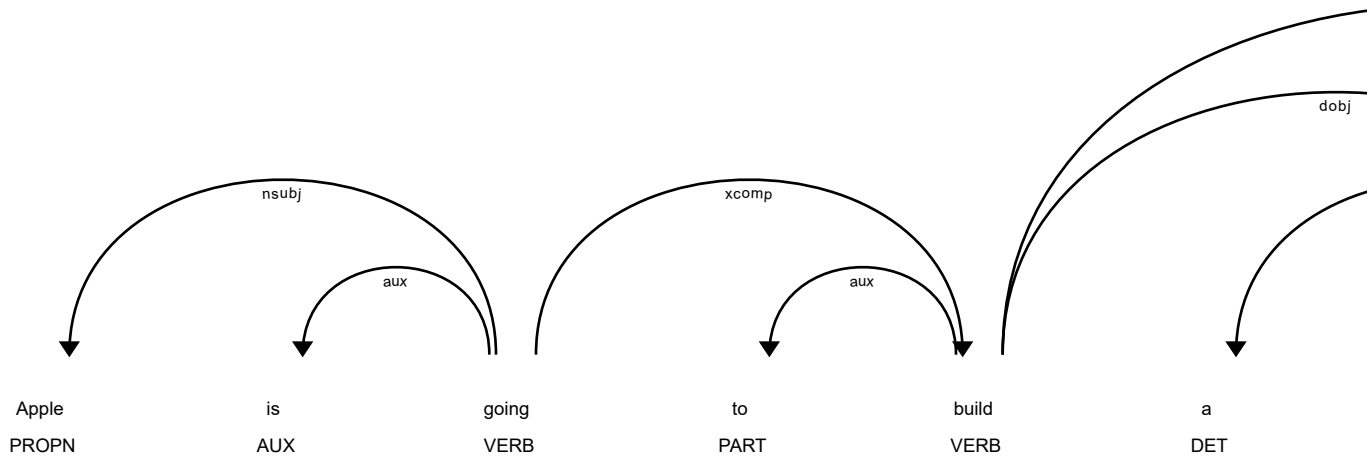
```
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))
```

```
stop_words
```

```
{'a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
 'been',
 'before',
 'being',
 'below',
 'between',
 'both',
 'but',
 'by',
 'can',
 'couldn',
 "couldn't",
 'd',
 'did',
 'didn',
 "didn't",
 'do',
 'does',
 'doesn',
 "doesn't",
 'doing',
 'don',
 "don't",
 'down',
 'during',
 'each',
 'few',
 'for',
 'from',
 'further',
 'had',
 'hadn',
 "hadn't",
 'has',
 'hasn',
 "hasn't",
 'have',
 'haven',
 "haven't",
 'having',
```

```
#Dependancy Parcing

from spacy import displacy
doc = nlp("Apple is going to build a U.K. Factory for $5 Million")
```

```
displacy.render(doc, style="dep", jupyter=True)
```



Stemming and Lemmetization

Stemming follows rule based approach Lemmetization is a word mapped corpus which is trained

```
text = "I studied artificial intelligence and then meeting Mr.Virat Tommorrow in a meeting"
```

```
from nltk.stem.porter import *
```

```
stemmer = PorterStemmer()
for word in text.split():
    print(word,": ",stemmer.stem(word))
```

```
I :  i
studied :  studi
artificial :  artifici
intelligence :  intellig
and :  and
then :  then
meeting :  meet
Mr.Virat :  mr.virat
Tommorrow :  tommorrow
in :  in
a :  a
meeting :  meet
```

```
doc = nlp(text)
```

```
for token in doc:
    print(token,":",token.pos_ ," : " , token.lemma_)
```

```
I : PRON  :  I
studied : VERB  :  study
artificial : ADJ  :  artificial
intelligence : NOUN  :  intelligence
and : CCONJ  :  and
then : ADV  :  then
meeting : VERB  :  meet
Mr. : PROPN  :  Mr.
Virat : PROPN  :  Virat
Tommorrow : PROPN  :  Tommorrow
in : ADP  :  in
a : DET  :  a
meeting : NOUN  :  meeting
```