7. NLP – Spacy library

Contents

6. NLP – Spacy library

1. Introduction

- ✓ There are mainly 3 types of data we have
  - Structured data
  - Semi structured data
  - Unstructured data

- ✓ Today more than 80% of the data available as Unstructured Data.
- ✓ Unstructured data is a data which cannot be represented in tabular format
  - Texts,
  - Videos,
  - Images

- ✓ We can use Natural Language processing to process and interpret the unstructured data
- ✓ NLP applications,
  - Sentiment analysis
  - Speech recognition
  - Text Classification
  - Machine Translation
  - Semantic Search
  - News/article Summarization
  - Answering Questions

- ✓ Live examples,
  - Google Assistant.
  - Amazon Echo.
  - ChatBot,
  - Language Translations,
  - Article summarization and so on.

2. Install and Load Main Python Libraries for NLP

- ✓ We do have different python libraries to work with NLP.
  - Nltk
  - Spacy
  - Genism
  - transformers

2.1. Installation

- ✓ Open a command prompt and run below commands

  - pip install nltk
  - pip install spacy
  - pip install gensim
  - pip install transformers

Note: Run below command

- ✓ python -m spacy download en_core_web_sm

3. Text Pre-processing

- ✓ The raw text data also called as text corpus, it has a lot of noise.
- ✓ Raw text having punctuation, special symbols, stop words & etc
- ✓ From the raw data we need to process the text by using nlp techniques.
- ✓ A text can be converted into nlp object of spacy.

Program Name: spacy example
demo1.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

raw_text = 'NLP is a very powerful tool'

text_doc = nlp(raw_text)

print(text_doc)
print(type(text_doc))
```

Output

```
NLP is a very powerful tool
<class 'spacy.tokens.doc.Doc'>
```

4. What is Tokenization?

- ✓ The process of extracting tokens from a text file/document is referred as tokenization.

Program Name: Tokenization using spacy
demo2.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

raw_text = 'NLP is a very powerful tool'

text_doc = nlp(raw_text)

print(text_doc)
print()
for word in text_doc:
        print(word.text)
```

Output

```
NLP is a very powerful tool

NLP
is
a
very
powerful
tool
```

Program Name: Checking every token either it is alphabet or not by using spacy
demo3.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

mixed_text = 'My salary is $1000 dollars'
mixed_text_doc = nlp(mixed_text)

for word in mixed_text_doc:
        print(word.text.ljust(10), word.is_alpha)
```

Output

```
My         True
salary     True
is         True
$          False
1000       False
dollars    True
```

Program Name: Checking every token alpha, stop word and punctuation
demo4.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

text='My salary is $1000 dollars'
text_doc=nlp(text)


print("Text",'\t\t', "Alpha", "Stop", "Punct")

for word in text_doc:
        print(word.text.ljust(15), ':', word.is_alpha, word.is_stop,
        word.is_punct)
```

Output

```
Text             Alpha Stop Punct
My              : True True False
salary          : True False False
is              : True True False
$               : False False False
1000            : False False False
dollars         : True False False
```

Program Name: Checking tokens and count of tokens by using spacy
demo5.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

raw_text= 'Amazon Alexa, also known simply as Alexa, is a virtual assistant AI technology developed by Amazon, first used in the Amazon Echo smart speakers developed by Amazon Lab126. It is capable of voice interaction, music playback, making to-do lists, setting alarms, streaming podcasts, playing audio books, and providing weather, traffic, sports, and other real-time information, such as news. Alexa can also control several smart devices using itself as a home automation system. Users are able to extend the Alexa capabilities by installing "skills" additional functionality developed by third-party vendors, in other settings more commonly called apps such as weather programs and audio features. Most devices with Alexa allow users to activate the device using a wake-word (such as Alexa or Amazon); other devices (such as the Amazon mobile app on iOS or Android and Amazon Dash Wand) require the user to push a button to activate Alexa listening mode, although, some phones also allow a user to say a command, such as "Alexa" or "Alexa wake". Currently, interaction and communication with Alexa are available only in English, German, French, Italian, Spanish, Portuguese, Japanese, and Hindi. '

text_doc = nlp(raw_text)

token_count = 0

for word in text_doc:
        print(word.text)
        token_count = token_count  + 1

print("Token count is:", token_count)
```

Output

```
Amazon
Alexa
,
also
known
simply
as
```

```
Alexa
,
is
a
.....

Token count is: 235
```

Note

- ✓ So, we have a lot of tokens here. The stop words like ‘it’, ’was’ ’that’, ’to’…, so on.
- ✓ These words will not give us much information, especially for models.
- ✓ Punctuations also will not provide much info
  - While dealing with large text files, the stop words and punctuations will be repeated at high levels, misguiding us to think they are important.
- ✓ So, it is necessary to filter out the stop words.

5. Stop words in spacy library

- ✓ Spacy has an inbuilt list of stop words.

Program Name: Checking stop words by using spacy
demo6.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

stopwords = spacy.lang.en.stop_words.STOP_WORDS

list_stopwords = list(stopwords)

print(list_stopwords)
```

Output

```
['re', 'through', '’ll', 'besides', 'in', 'per', 'along', 'rather', 'mostly', 'ourselves', 'around', 'f
urther', 'always', 'same', 'among', 'that', 'we', 'yet', 'n’t', 'hereupon', 'own', 'have', 'must', 'the
refore', 'except', 'into', 'some', 'too', 'her', 'could', 'are', 'but', 'hereby', "'s", '’ve', 'least',
 'indeed', 'over', 'using', 'do', 'whenever', 'nowhere', 'therein', 'call', 'wherein', 'your', 'meanwhi
le', 'due', '’s', 'go', 'part', 'no', 'themselves', 'onto', 'moreover', 'almost', 'without', 'am', 'sid
e', 'a', 'make', 'anyway', 'just', 'then', 'while', 'been', 'every', 'under', 'will', 'seem', 'none', '
latterly', 'about', 'please', 'say', 'beside', 'see', 'forty', 'unless', 'ca', 'whereupon', 'everyone',
 'become', 'had', 'why', 'though', 'eleven', 'take', 'be', 'after', 'via', 'than', 'these', 'where', 'a
nyhow', 'each', 'first', 'made', 'many', 'most', 'has', 'name', 'nobody', 'amongst', 'one', 'thru', 'us
ed', 'whence', 'very', 'everything', 'cannot', 'yourselves', 'somewhere', 'would', 'of', 'four', 'at',
'elsewhere', 'may', 'last', 'before', 'much', 'next', 'hence', 'throughout', 'for', 'twelve', 'you', 's
till', 'either', '’m', 'when', 'top', 'were', 'wherever', 'with', 'alone', 'eight', 'becomes', 'front',
 'an', 'again', 'fifty', 'whether', "'ll", 'latter', 'against', 'became', 'and', 'on', 'anyone', 'anyth
ing', 'sometime', 'thereupon', 'i', 'various', 'up', 'thence', "n't", 'here', 'less', 'seeming', '‘s',
'itself', 'out', 'sometimes', 'thereby', 'third', 'to', 'quite', 'together', '‘ve', 'yours', 'often', '
nor', 'move', 'twenty', 'my', 'also', 'all', 'keep', 'doing', 'however', 'hundred', 'back', 'since', 'o
thers', 'something', 'those', 'well', 'because', 'whereby', 'hereafter', 'himself', 'me', 'full', 'hers
elf', 'myself', 'whatever', 'seemed', 'even', 'within', 'behind', 'n‘t', 'it', 'formerly', 'perhaps', '
down', 'becoming', 'them', 'upon', 'they', 'afterwards', 'ten', 'whoever', 'us', "'ve", 'she', 'otherwi
se', 'beyond', 'ever', 'which', 'once', 'was', 'so', 'whom', '‘m', 'bottom', 'neither', 'any', 'anywher
e', 'else', 'his', 'now', 'several', '’re', 'few', "'d", 'above', 'there', 'by', 'off', 'is', 'serious'
, 'if', 'get', 'sixty', 'during', 'seems', 'everywhere', 'regarding', 'namely', '‘re', 'give', '’d', 'o
ther', '‘d', 'him', 'thereafter', 'towards', 'between', 'or', 'although', 'does', 'below', 'noone', 'be
```

Program Name: Removing **stop words** and **punctuations** from text using spacy
demo7.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

raw_text = "Hello good!!! morning how are you, today climate is very cool"

text_doc = nlp(raw_text)

final = [
                token
                for token in text_doc
                if not token.is_stop and not token.is_punct
]

print("Actual text:", raw_text)

for token in final:
        print(token)
```

Output

```
Actual text: Hello good!!! morning how are you, today climate is very cool
Hello
good
morning
today
climate
cool
```

6. Stemming

- ✓ General words are,
  - ‘calculator’,
  - ‘calculating’,
  - ‘Calculation’.
- ✓ We know that these words are not very distinct from each other.
- ✓ It can be achieved through stemming.
- ✓ Stemming means reducing a word to its ‘root form’.
- ✓ We can implement this by using,
  - PorterStemmer predefined class

Program Name: Stemming example
demo8.py

```
from nltk.stem import PorterStemmer

ps = PorterStemmer()

words = ['dance ','dances', 'dancing ','danced']

for word in words:
        print(word,'------->', ps.stem(word))
```

Output

```
dance  -------> dance
dances -------> danc
dancing  -------> dancing
danced -------> danc
```

Note

- ✓ You can observe that some words were reduced to the base stems ‘danc’
- ✓ But, the word ‘danc’ is not semantically correct.
- ✓ Stemming may give us root words that are not present in the dictionary.
- ✓ How to overcome this?
  - That’s where Lemmatization comes to rescue

7. Lemmatization

- ✓ It is similar to stemming, except that the root word is correct and always meaningful.
- ✓ There are different ways to perform lemmatization.
- ✓ We will learn lemmatization by using nltk and spacy in below examples.

8. Lemmatization using spacy

- ✓ In spacy, the token object has an attribute .lemma_ which allows you to access the lemmatized version of that token.

Program Name: Lemmatization using spacy
demo9.py

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

text = 'dance dances dancing danced'
text_doc = nlp(text)

for token in text_doc:
        print(token.text,'-------', token.lemma_)
```

Output

```
dance ------- dance
dances ------- dance
dancing ------- dance
danced ------- dance
```

Note

- ✓ Here, all words are reduced to ‘dance’ which is meaningful and just as required.
- ✓ It is highly preferred over stemming.

9. Word Frequency Analysis

- ✓ Once we removed stop words and applied lemmatization then the next important step is, we need analyse for further information about the text data.
- ✓ If any specific word repeated more times then we need to focus on that well.
- ✓ So, the words which occur more frequently those are very key concepts
- ✓ So, we need to store all tokens with their frequencies separately to analyse well

10. Counter predefined class

- ✓ We can use Counter to get the frequency of each token.
- ✓ If you provide a list to the Counter it returns a dictionary of all elements with their frequency as values.

Program Name: Counter example
demo10.py

```
import collections
from collections import Counter

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


data = 'It is my birthday today. I could not have a birthday party. I felt sad'
data_doc = nlp(data)

list_of_tokens = [token.text for token in data_doc if not token.is_stop and not token.is_punct]

token_frequency = Counter(list_of_tokens)
print(token_frequency)
```

Output

```
Counter({'birthday': 2, 'today': 1, 'party': 1, 'felt': 1, 'sad': 1})
```

Note

- ✓ From above output, the word ‘birthday’ is most common.
- ✓ So, it gives us an idea that the text is about a birthday.
- ✓ Let us try with another example with more larger data
- ✓ For example,
  - o If we have a text data about a particular place, and you want to know the important factors.

Program Name: Counter example on large text
demo11.py

```
import collections
from collections import Counter

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

longtext = 'Gangtok is a city, municipality, the capital and the largest town of
the Indian state of Sikkim. It is also the headquarters of the East Sikkim
district. Gangtok is in the eastern Himalayan range, at an elevation of 1,650.
The towns population of 100000 are from different ethnicities such as Bhutia,
Lepchas and Indian Gorkhas. Within the higher peaks of the Himalaya and
with a year-round mild temperate climate, Gangtok is at the centre of Sikkims
tourism industry. Gangtok rose to prominence as a popular Buddhist
pilgrimage site after the construction of the Enchey Monastery in 1840. In
1894, the ruling Sikkimese Chogyal, Thutob Namgyal, transferred the capital
to Gangtok. In the early 20th century, Gangtok became a major stopover on
the trade route between Lhasa in Tibet and cities such as Kolkata (then
Calcutta) in British India. After India won its independence from Britain in
1947, Sikkim chose to remain an independent monarchy, with Gangtok as its
capital. In 1975, after the integration with the union of India, Gangtok was
made Indias 22nd state capital. Like the rest of Sikkim, not much is known
about the early history of Gangtok. The earliest records date from the
construction of the hermitic Gangtok monastery in 1716.[7] Gangtok
remained a small hamlet until the construction of the Enchey Monastery in
1840 made it a pilgrimage center. It became the capital of what was left of
Sikkim after an English conquest in the mid-19th century in response to a
hostage crisis. After the defeat of the Tibetans by the British, Gangtok
became a major stopover in the trade between Tibet and British India at the
end of the 19th century. Most of the roads and the telegraph in the area
were built during this time. In 1894, Thutob Namgyal, the Sikkimese monarch
under British rule, shifted the capital from Tumlong to Gangtok, increasing
the citys importance. A new grand palace along with other state buildings
was built in the new capital. Following India independence in 1947, Sikkim
became a nation-state with Gangtok as its capital. Sikkim came under the
suzerainty of India, with the condition that it would retain its independence,
by the treaty signed between the Chogyal and the then Indian Prime Minister
Jawaharlal Nehru.[9] This pact gave the Indians control of external affairs on
behalf of Sikkimese. Trade between India and Tibet continued to flourish
through the Nathula and Jelepla passes, offshoots of the ancient Silk Road
near Gangtok. These border passes were sealed after the Sino-Indian War in
1962, which deprived Gangtok of its trading business.[10] The Nathula pass
```

```
was finally opened for limited trade in 2006, fuelling hopes of economic
boomIn 1975, after years of political uncertainty and struggle, including riots,
the monarchy was abrogated and Sikkim became India twenty-second state,
with Gangtok as its capital after a referendum. Gangtok has witnessed annual
landslides, resulting in loss of life and damage to property. The largest
disaster occurred in June 1997, when 38 were killed and hundreds of
buildings were destroyed'

long_text = nlp(longtext)

list_of_tokens = [token.text for token in long_text if not token.is_stop and
not token.is_punct]

token_frequency = Counter(list_of_tokens)
print(token_frequency)
```

Output

```
Counter({'Gangtok': 18, 'capital': 9, 'Sikkim': 8, 'India': 8, 'state': 5, 'Indian': 4,
'British': 4, 'construction': 3, 'Sikkimese': 3, 'century': 3, 'trade': 3, 'Tibet': 3,
'independence': 3, 'largest': 2, 'pilgrimage': 2, 'Enchey': 2, 'Monastery': 2,
'1840': 2, '1894': 2, 'Chogyal': 2, 'Thutob': 2, 'Namgyal': 2, 'early': 2, 'major': 2,
'stopover': 2, '1947': 2, 'monarchy': 2, '1975': 2, 'built': 2, 'new': 2, 'buildings':
2, 'Nathula': 2, 'passes': 2, 'city': 1, 'municipality': 1, 'town': 1, 'headquarters':
1, 'East': 1, 'district': 1, 'eastern': 1, 'Himalayan': 1, 'range': 1, 'elevation': 1,
'1,650': 1, 'towns': 1, 'population': 1, '100000': 1, 'different': 1, 'ethnicities': 1,
'Bhutia': 1, 'Lepchas': 1, 'Gorkhas': 1, 'higher': 1, 'peaks': 1, 'Himalaya': 1,
'year': 1, 'round': 1, 'mild': 1, 'temperate': 1, 'climate': 1, 'centre': 1, 'Sikkims':
1, 'tourism': 1, 'industry': 1, 'rose': 1, 'prominence': 1, 'popular': 1, 'Buddhist':
1, 'site': 1, 'ruling': 1, 'transferred': 1, '20th': 1, 'route': 1, 'Lhasa': 1, 'cities': 1,
'Kolkata': 1, 'Calcutta': 1, 'won': 1, 'Britain': 1, 'chose': 1, 'remain': 1,
'independent': 1, 'integration': 1, 'union': 1, 'Indias': 1, '22nd': 1, 'Like': 1,
'rest': 1, 'known': 1, 'history': 1, 'earliest': 1, 'records': 1, 'date': 1, 'hermitic':
1, 'monastery': 1, '1716.[7': 1, 'remained': 1, 'small': 1, 'hamlet': 1, 'center': 1,
'left': 1, 'English': 1, 'conquest': 1, 'mid-19th': 1, 'response': 1, 'hostage': 1,
'crisis': 1, 'defeat': 1, 'Tibetans': 1, 'end': 1, '19th': 1, 'roads': 1, 'telegraph': 1,
'area': 1, 'time': 1, 'monarch': 1, 'rule': 1, 'shifted': 1, 'Tumlong': 1, 'increasing':
1, 'citys': 1, 'importance': 1, 'grand': 1, 'palace': 1, 'Following': 1, 'nation': 1,
'came': 1, 'suzerainty': 1, 'condition': 1, 'retain': 1, 'treaty': 1, 'signed': 1,
'Prime': 1, 'Minister': 1, 'Jawaharlal': 1, 'Nehru.[9': 1, 'pact': 1, 'gave': 1,
'Indians': 1, 'control': 1, 'external': 1, 'affairs': 1, 'behalf': 1, 'Trade': 1,
'continued': 1, 'flourish': 1, 'Jelepla': 1, 'offshoots': 1, 'ancient': 1, 'Silk': 1,
'Road': 1, 'near': 1, 'border': 1, 'sealed': 1, 'Sino': 1, 'War': 1, '1962': 1,
'deprived': 1, 'trading': 1, 'business.[10': 1, 'pass': 1, 'finally': 1, 'opened': 1,
'limited': 1, '2006': 1, 'fuelling': 1, 'hopes': 1, 'economic': 1, 'boomIn': 1,
'years': 1, 'political': 1, 'uncertainty': 1, 'struggle': 1, 'including': 1, 'riots': 1,
```

```
'abrogated': 1, 'second': 1, 'referendum': 1, 'witnessed': 1, 'annual': 1,
'landslides': 1, 'resulting': 1, 'loss': 1, 'life': 1, 'damage': 1, 'property': 1,
'disaster': 1, 'occurred': 1, 'June': 1, '1997': 1, '38': 1, 'killed': 1, 'hundreds': 1,
'destroyed': 1})
```

Note

- ✓ As you can see, as the length or size of text data increases, it is difficult to analyse frequency of all tokens.
- ✓ So, you can print the n most common tokens using most_common function of Counter.
- ✓ see the below example on how to print the 6 most frequent words of the text

Program Name: Most common words from text
demo12.py

```
import collections
from collections import Counter

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

longtext = 'Gangtok is a city, municipality, the capital and the largest town of
the Indian state of Sikkim. It is also the headquarters of the East Sikkim
district. Gangtok is in the eastern Himalayan range, at an elevation of 1,650.
The towns population of 100000 are from different ethnicities such as Bhutia,
Lepchas and Indian Gorkhas. Within the higher peaks of the Himalaya and
with a year-round mild temperate climate, Gangtok is at the centre of Sikkims
tourism industry. Gangtok rose to prominence as a popular Buddhist
pilgrimage site after the construction of the Enchey Monastery in 1840. In
1894, the ruling Sikkimese Chogyal, Thutob Namgyal, transferred the capital
to Gangtok. In the early 20th century, Gangtok became a major stopover on
the trade route between Lhasa in Tibet and cities such as Kolkata (then
Calcutta) in British India. After India won its independence from Britain in
1947, Sikkim chose to remain an independent monarchy, with Gangtok as its
capital. In 1975, after the integration with the union of India, Gangtok was
made Indias 22nd state capital. Like the rest of Sikkim, not much is known
about the early history of Gangtok. The earliest records date from the
construction of the hermitic Gangtok monastery in 1716.[7] Gangtok
remained a small hamlet until the construction of the Enchey Monastery in
1840 made it a pilgrimage center. It became the capital of what was left of
Sikkim after an English conquest in the mid-19th century in response to a
hostage crisis. After the defeat of the Tibetans by the British, Gangtok
became a major stopover in the trade between Tibet and British India at the
end of the 19th century. Most of the roads and the telegraph in the area
were built during this time. In 1894, Thutob Namgyal, the Sikkimese monarch
under British rule, shifted the capital from Tumlong to Gangtok, increasing
the citys importance. A new grand palace along with other state buildings
was built in the new capital. Following India independence in 1947, Sikkim
became a nation-state with Gangtok as its capital. Sikkim came under the
suzerainty of India, with the condition that it would retain its independence,
by the treaty signed between the Chogyal and the then Indian Prime Minister
Jawaharlal Nehru.[9] This pact gave the Indians control of external affairs on
behalf of Sikkimese. Trade between India and Tibet continued to flourish
through the Nathula and Jelepla passes, offshoots of the ancient Silk Road
near Gangtok. These border passes were sealed after the Sino-Indian War in
1962, which deprived Gangtok of its trading business.[10] The Nathula pass
was finally opened for limited trade in 2006, fuelling hopes of economic
```

```
boomIn 1975, after years of political uncertainty and struggle, including riots,
the monarchy was abrogated and Sikkim became India twenty-second state,
with Gangtok as its capital after a referendum. Gangtok has witnessed annual
landslides, resulting in loss of life and damage to property. The largest
disaster occurred in June 1997, when 38 were killed and hundreds of
buildings were destroyed'

long_text = nlp(longtext)

list_of_tokens = [
        token.text
        for token in long_text
        if not token.is_stop and not token.is_punct
]

token_frequency = Counter(list_of_tokens)

most_frequent_tokens = token_frequency.most_common(6)
print(most_frequent_tokens)
```

Output

```
[('Gangtok', 18), ('capital', 9), ('Sikkim', 8), ('India', 8), ('state', 5), ('Indian', 4)]
```

Note

- ✓ We can see that the keywords are gangtok , sikkkim, Indian and so on.
- ✓ It gives us an idea of the text to start with.

11. Part of Speech Tagging

- ✓ Each word has its own role in a sentence.
- ✓ For example,
  - ’Daniel is dancing’.
  - Daniel is the person or ‘Noun’ and dancing is the action performed by him.
  - So it is a ‘Verb’.
  - Likewise, each word can be classified.
  - This is referred as POS or Part of Speech Tagging.

12. Parts of Speech

- ✓ Noun
- ✓ Verb
- ✓ Adjective
- ✓ Pronoun
- ✓ Adverb
- ✓ Preposition
- ✓ Conjunction
- ✓ Interjection and so on.

13. Pos by using spacy

- ✓ POS can be implemented using both spaCy and nltk.
- ✓ First, let us see the method using spaCy
- ✓ In spaCy, the POS tags are present in the attribute of Token object.
- ✓ We can access the POS tag of particular token through the token.pos_ attribute.

Program Name: Pos example
demo13.py

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

text = 'Daniel is singing loudly and his roommates are enjoying too'
text_doc = nlp(text)

for word in text_doc:
        print(word.text.ljust(10), '-----', word.pos_)
```

Output

```
Nireekshan ----- PROPN
is         ----- AUX
singing    ----- VERB
loudly     ----- ADV
and        ----- CCONJ
his        ----- DET
roommates  ----- NOUN
are        ----- AUX
enjoying   ----- VERB
too        ----- ADV
```

covid.txt

Even as the UK became the first country to grant emergency approval to Pfizer-BioNTech's Covid-19 vaccine, the US firm said it is "committed" to engaging with the Indian government to "explore opportunities" to roll it out here. The vaccine will be a challenge for India as it requires storage at -70 degrees, experts say.

Pfizer spokeswoman Roma Nair told TOI: "We are committed to advance our dialogue with the Indian government. We are working with governments across the world to understand the infrastructure requirements of each country and we have logistical plans in place. We are confident the rollout can be managed in India."

Program Name: Preprocessing the text by using spacy
demo14.py

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

with open('covid.txt') as file:
        robot_text = file.read()

robot_doc = nlp(robot_text)

robot_doc = [
        word
        for word in robot_doc
        if not word.is_stop and not word.is_punct
]

print(robot_doc)
```

Output

[UK, country, grant, emergency, approval, Pfizer, BioNTech, Covid-19, vaccine, firm, said, committed, engaging, Indian, government, explore, opportunities, roll, vaccine, challenge, India, requires, storage, -70, degrees, experts,
, Pfizer, spokeswoman, Roma, Nair, told, TOI, committed, advance, dialogue, Indian, government, working, governments, world, understand, infrastructure, requirements, country, logistical, plans, place, confident, rollout, managed, India]

Program Name: List of nouns and verbs
demo15.py

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

with open('covid.txt') as file:
        robot_text = file.read()

robot_doc = nlp(robot_text)

robot_doc = [
        word
        for word in robot_doc
        if not word.is_stop and not word.is_punct
]

nouns = []
verbs = []

for word in robot_doc:
        if word.pos_ == 'NOUN':
                nouns.append(word)
        if word.pos_ == 'VERB':
                verbs.append(word)

print('List of Nouns in the text:', nouns)
print()
print('List of verbs in the text:', verbs)
```

Output

```
List of Nouns in the text:
[country, emergency, approval, vaccine, firm, government, opportunities,
vaccine, challenge, storage, degrees, experts, spokeswoman, dialogue,
government, governments, world, infrastructure, requirements, country,
plans, place, rollout]

List of verbs in the text:
[grant, said, committed, engaging, explore, roll, requires, told, advance,
working, understand, managed]
```

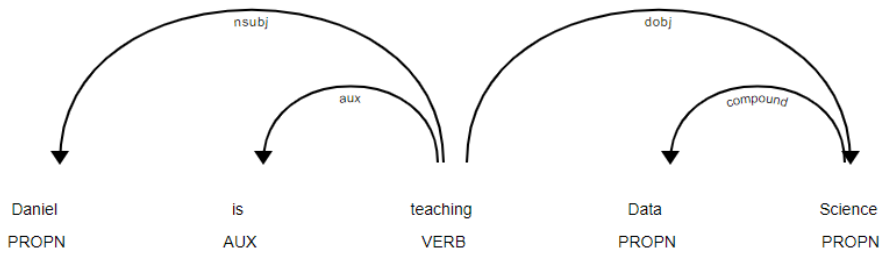
Program Name

We can use displacy function for display.
demo16.py

```
from spacy import displacy

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

text = "Daniel is teaching Data Science"

doc = nlp(text)
displacy.render(doc)
```

Note

Execute above example in jupyter to see output image



14. Named Entity Recognition

Named Entity Reorganization

- ✓ Suppose you have a collection of news articles text data.
- ✓ From these we can capture,
  - companies/organization names
  - People names
- ✓ By using NER topic, we can do this

15. What is NER?

- ✓ NER is the technique of identifying named entities in the text corpus and assigning them pre-defined categories such as ' person names' , ' locations' ,'organizations', etc..
- ✓ It is a very useful method especially in the field of classification problems and search engine optimizations.

16. NER by using nltk and spacy

- ✓ NER can be implemented through both nltk and spacy'.

17. NER with spacy

- ✓ Spacy is highly flexible and advanced library.
- ✓ Named entity recognition through spacy is easy.

18. The few common labels are:

- ✓ 'ORG' : companies, organizations, etc.
- ✓ 'PERSON' : names of people
- ✓ 'GPE' : countries, states, etc.
- ✓ 'PRODUCT' : vehicles, food products and so on
- ✓ 'LANGUAGE' : Names of different languages

- ✓ There are many other labels too.

Program Name: Print all the named entities
demo20.py

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

sentence=' The building is located at London. It is the headquarters of
Google. Mark works there. He speaks English'

doc=nlp(sentence)

for entity in doc.ents:
        print(entity.text,'--', entity.label_)
```

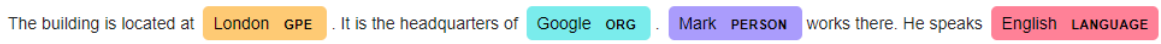
Output

```
London -- GPE
Google -- ORG
Mark -- PERSON
English -- LANGUAGE
```

Program Name: Visualize all the named entities
demo21.py

```
import spacy

nlp = spacy.load('en_core_web_sm')

sentence=' The building is located at London. It is the headquarters of Google. Mark works there. He speaks English'

doc=nlp(sentence)

displacy.render(doc, style='ent', jupyter=True)
```

Output



Use case - 1

- ✓ Let us take the text data of collection of news headlines.
- ✓ Can we get list of all names that have occurred in the news?

Program Name: Display only person names from the text
demo22.py

```
from spacy import displacy
import spacy

nlp = spacy.load('en_core_web_sm')

news_articles = 'Honoured to serve India, Naredra Modi, 68, wrote in a
Twitter post that popped up on the micro blogging site as the ceremony
started at 7 p.m. before an audience of 8,000 people, who included United
Progressive Alliance chairperson Sonia Gandhi and her son and Congress
president Rahul Gandhi, both seated on the front row. Shah, 54, whose
inclusion in the cabinet had been much speculated upon, was administered
the oaths after Naredra Modi and Rajnath Singh, indicating the order of
seniority in the new government , which took office exactly a week after
results from the 17th general elections were declared'

news_doc=nlp(news_articles)

list_of_people=[]

for token in news_doc:
        if token.ent_type_=='PERSON':
                list_of_people.append(token.text)

print(list_of_people)
```

Output

```
['Sonia', 'Gandhi', 'Rahul', 'Gandhi', 'Naredra', 'Modi', 'Rajnath', 'Singh']
```