Introduction

Natural language processing is a set of methods to make human language understable to computer. NLP plays an important role in our lives. From automatic language transition in social media(like twitter, facebook etc.), text classification to keep our mail safe plays an important role.It is a diverse set of applications including statistics, logis, ideas, linguistics etc.

The goal of NLP is “to accomplish human-like language processing”. A full NLU System would be able to:

1. Paraphrase an input text
2. Translate the text into another language
3. Answer questions about the contents of the text
4. Draw inferences from the text

Some NLP applications

The following are some useful applications:

- sentiment analysis
- optical character recognition (OCR)
- speech recognition
- chatbot
- document classification
- document clustering
- autocomplete

Understanding Why NLP?

We as a human do perform natural language processing, even though we are not perfect. We often mix the meaning of different words, misunderstand one thing from another.

Consider: Can you help me with the Can?

In the above sentence there are two “can” words, but both of them have different meanings. The first “can” word is used for question formation and the second “can” word at the end of the sentence represents a container.

NLP libraries

1. NLTK (Natural Language Toolkit)

   NLP with Python provides a practical introduction to programming for language processing. The NLTK Python framework is generally used as an education and research tool.

Features:

- Tokenization
- Part Of Speech tagging (POS)
- Named Entity Recognition (NER)
- Classification

Use-cases:

- Recommendation systems
- Sentiment analysis
- Creating Chatbots

2. spaCy

   spaCy is an open-source natural language processing Python library designed to be fast and production-ready. spaCy focuses on providing software for production usage.

Features:

- Sentiment analysis
- Dependency parsing
- Word vectors

Use-cases:

- Autocomplete and autocorrect
- Analysing reviews

3. TextBlob

   It is a Python library designed for processing textual data.

Features:

- Part-of-Speech tagging
- Noun phrase extraction
- Language translation
- Wordnet integration

Use-cases:

- Spelling Correction
- Translation and Language Detection

Text preprocessing with python libraries

```
! pip install nltk # install NLTK
! python -m nltk.downloader popular #installing NLTK Data
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/pub
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages (3.7)
Requirement already satisfied: tqdm in /usr/local/lib/python3.7/dist-packages (from nlt
Requirement already satisfied: click in /usr/local/lib/python3.7/dist-packages (from nl
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: joblib in /usr/local/lib/python3.7/dist-packages (from n
/usr/lib/python3.7/runpy.py:125: RuntimeWarning: 'nltk.downloader' found in sys.modules
  warn(RuntimeWarning(msg))
[nltk_data] Downloading collection 'popular'
[nltk_data]    |
[nltk_data]    | Downloading package cmudict to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/cmudict.zip.
[nltk_data]    | Downloading package gazetteers to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/gazetteers.zip.
[nltk_data]    | Downloading package genesis to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/genesis.zip.
[nltk_data]    | Downloading package gutenberg to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/gutenberg.zip.
[nltk_data]    | Downloading package inaugural to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/inaugural.zip.
[nltk_data]    | Downloading package movie_reviews to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/movie_reviews.zip.
[nltk_data]    | Downloading package names to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/names.zip.
[nltk_data]    | Downloading package shakespeare to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/shakespeare.zip.
[nltk_data]    | Downloading package stopwords to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/stopwords.zip.
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/treebank.zip.
[nltk_data]    | Downloading package twitter_samples to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/twitter_samples.zip.
[nltk_data]    | Downloading package omw to /root/nltk_data...
[nltk_data]    | Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]    | Downloading package wordnet to /root/nltk_data...
[nltk_data]    | Downloading package wordnet2021 to /root/nltk_data...
[nltk_data]    | Downloading package wordnet31 to /root/nltk_data...
[nltk_data]    | Downloading package wordnet_ic to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/wordnet_ic.zip.
[nltk_data]    | Downloading package words to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/words.zip.
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
```

```
[nltk_data]    |   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data]    | Downloading package punkt to /root/nltk_data...
[nltk_data]    |   Unzipping tokenizers/punkt.zip.
[nltk_data]    | Downloading package snowball_data to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection popular
```

```
import nltk
nltk.download()
```

```
NLTK Downloader
---------------------------------------------------------------------------
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
Downloader> l

Packages:
  [ ] abc..................... Australian Broadcasting Commission 2006
  [ ] alpino.................. Alpino Dutch Treebank
  [*] averaged_perceptron_tagger Averaged Perceptron Tagger
  [ ] averaged_perceptron_tagger_ru Averaged Perceptron Tagger (Russian)
  [ ] basque_grammars..... Grammars for Basque
  [ ] biocreative_ppi..... BioCreAtIvE (Critical Assessment of Information
                           Extraction Systems in Biology)
  [ ] bllip_wsj_no_aux.... BLLIP Parser: WSJ Model
  [ ] book_grammars....... Grammars from NLTK Book
  [ ] brown............... Brown Corpus
  [ ] brown_tei........... Brown Corpus (TEI XML Version)
  [ ] cess_cat............ CESS-CAT Treebank
  [ ] cess_esp............ CESS-ESP Treebank
  [ ] chat80.............. Chat-80 Data Files
  [ ] city_database....... City Database
  [*] cmudict............. The Carnegie Mellon Pronouncing Dictionary (0.6)
  [ ] comparative_sentences Comparative Sentence Dataset
  [ ] comtrans............ ComTrans Corpus Sample
  [ ] conll2000........... CONLL 2000 Chunking Corpus
  [ ] conll2002........... CONLL 2002 Named Entity Recognition Corpus
Hit Enter to continue:
  [ ] conll2007........... Dependency Treebanks from CoNLL 2007 (Catalan
                           and Basque Subset)
  [ ] crubadan............ Crubadan Corpus
  [ ] dependency_treebank. Dependency Parsed Treebank
  [ ] dolch............... Dolch Word List
  [ ] europarl_raw........ Sample European Parliament Proceedings Parallel
                           Corpus
  [ ] extended_omw........ Extended Open Multilingual WordNet
  [ ] floresta............ Portuguese Treebank
  [ ] framenet_v15........ FrameNet 1.5
  [ ] framenet_v17........ FrameNet 1.7
```

```
  [*] gazetteers..... Gazeteer Lists
  [*] genesis..... Genesis Corpus
  [*] gutenberg..... Project Gutenberg Selections
  [ ] ieer..... NIST IE-ER DATA SAMPLE
  [*] inaugural..... C-Span Inaugural Address Corpus
  [ ] indian..... Indian Language POS-Tagged Corpus
  [ ] jeita..... JEITA Public Morphologically Tagged Corpus (in
                 ChaSen format)
  [ ] kimmo..... PC-KIMMO Data Files
  [ ] knbc..... KNB Corpus (Annotated blog corpus)
Hit Enter to continue:
  [ ] large_grammars..... Large context-free and feature-based grammars
                 for parser comparison
  [ ] lin_thesaurus..... Lin's Dependency Thesaurus
  [ ] mac_morpho..... MAC-MORPHO: Brazilian Portuguese news text with
                 part-of-speech tags
  [ ] machado..... Machado de Assis -- Obra Completa
  [ ] masc_tagged..... MASC Tagged Corpus
```

Tokenize sentences and words

Splitting the text into sentences and words, is called tokenization. Tokenizing a text makes further analysis easier. Almost all text analysis applications start with this step.

```
text = "One small positive thought can change your whole day!"
```

```
#sentence tokenization
from nltk.tokenize import sent_tokenize
sentences = sent_tokenize(text)
print(len(sentences), 'sentences:', sentences)
```

```
1 sentences: ['One small positive thought can change your whole day!']
```

```
#word tokenization
from nltk.tokenize import word_tokenize
words = word_tokenize(text)
print(len(words), 'words:', words)
```

```
10 words: ['One', 'small', 'positive', 'thought', 'can', 'change', 'your', 'whole', 'day
```

Stopwords

These words are necessary in speech and writing but do not carry value in analysis.

Let filter out the stopwords

```
#printing total number of stopwords library includes
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
print(len(stop_words), "stopwords:", stop_words)
```

```
179 stopwords: ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you
```

```
text = "One small positive thought can change your whole day!"
from nltk.tokenize import word_tokenize
words = word_tokenize(text)
print(len(words), "in original text:", words)
```

```
10 in original text: ['One', 'small', 'positive', 'thought', 'can', 'change', 'your', 'w
```

```
#filtering out the stopword
words = [word for word in words if word not in stop_words]
print(len(words), "without stopwords:", words)
```

```
8 without stopwords: ['One', 'small', 'positive', 'thought', 'change', 'whole', 'day',
```

```
#printing all the puncuation marks
import string
punctuations = list(string.punctuation)
print(punctuations)
```

```
['!', '"', '#', '$', '%', '&', "'", '(', ')', '*', '+', ',', '-', '.', '/', ':', ';', '<
```

```
#removing puncuation
words = [word for word in words if word not in punctuations]
print(len(words), "words without stopwords and punctuations:", words)
```

```
7 words without stopwords and punctuations: ['One', 'small', 'positive', 'thought', 'cha
```

Parts of speech identification

NLTK has the ability to identify words' parts of speech (POS).

```
#preprocessing as did in the above steps
text = "One small positive thought can change your whole day!"
import nltk
```

```
import nltk
from nltk.tokenize import word_tokenize
words = word_tokenize(text)
```

```
#figuring out the POS
pos_tagged_text = nltk.pos_tag(words)
print(pos_tagged_text)
```

```
[('One', 'CD'), ('small', 'JJ'), ('positive', 'JJ'), ('thought', 'NN'), ('can', 'MD'), (
```