Tokenization

```
1 import nltk
2 nltk.download('popular')
```

```
[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]    |   Unzipping corpora/omw.zip.
[nltk_data]    | Downloading package omw-1.4 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/omw-1.4.zip.
[nltk_data]    | Downloading package wordnet to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/wordnet.zip.
[nltk_data]    | Downloading package wordnet2021 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/wordnet2021.zip.
[nltk_data]    | Downloading package wordnet31 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/wordnet31.zip.
[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
True
```

```
paragraph="""A true friend loves you unconditionally, understands you, but never judges
The friendship of Krishna and Sudama is a great example of true friendship.
A true friend is one who will always be there when you need someone.
He will leave all his important work, but will never leave you alone, especially in you
That is why it is said a friend in need is a friend indeed.
Difficult times are the best time to realize who your true friends are.
Blessed are the souls who have true friends. """
```

```
sentences=nltk.sent_tokenize(paragraph)
print(sentences,end=" ")
```

```
['A true friend loves you unconditionally, understands you, but never judges you and
```

```
words=nltk.word_tokenize(paragraph)
print(words)
```

```
['A', 'true', 'friend', 'loves', 'you', 'unconditionally', ',', 'understands', 'you',
```

Part Of Speech Tagging

```
from nltk.corpus import stopwords
```

```
stop_words = set(stopwords.words('english'))
```

```
for i in sentences:
    wordsList = nltk.word_tokenize(i)
    wordsList = [w for w in wordsList if not w in stop_words]
    tagged = nltk.pos_tag(wordsList)
    print(tagged)
```

```
[('A', 'DT'), ('true', 'JJ'), ('friend', 'NN'), ('loves', 'VBZ'), ('unconditionally',
[('The', 'DT'), ('friendship', 'NN'), ('Krishna', 'NNP'), ('Sudama', 'NNP'), ('great
[('A', 'DT'), ('true', 'JJ'), ('friend', 'NN'), ('one', 'CD'), ('always', 'RB'), ('ne
[('He', 'PRP'), ('leave', 'VBP'), ('important', 'JJ'), ('work', 'NN'), (',', ','), (
[('That', 'DT'), ('said', 'VBD'), ('friend', 'VBP'), ('need', 'MD'), ('friend', 'VB')
[('Difficult', 'NNP'), ('times', 'NNS'), ('best', 'JJS'), ('time', 'NN'), ('realize',
[('Blessed', 'VBN'), ('souls', 'NNS'), ('true', 'JJ'), ('friends', 'NNS'), ('.', '.')
```

Stop Word Removal

```
for i in range (len(sentences)):
    words=nltk.word_tokenize(sentences[i])
    words=[word for word in words if word not in set(stopwords.words('english'))]
    sentences[i]=' '.join(words)
print(sentences)
```

```
['A true friend loves unconditionally , understands , never judges always tries suppo
```

Stemming

```
from nltk.stem import PorterStemmer
stemmer=PorterStemmer()
for i in range (len(sentences)):
    words=nltk.word_tokenize(sentences[i])
    words=[stemmer.stem(word) for word in words if word not in set(stopwords.words('eng
    sentences[i]=' '.join(words)
print(sentences)
```

```
['A true friend love uncondit , understand , never judg alway tri support give good a
```

Lemmatization

```
from nltk.stem import WordNetLemmatizer
import re
```

```
sentences=nltk.sent_tokenize(paragraph)
lemmatizer=WordNetLemmatizer()
```

```
for i in range (len(sentences)):
    words=nltk.word_tokenize(sentences[i])
    words=[lemmatizer.lemmatize(word) for word in words if word not in set(stopwords.wo
    sentences[i]=' '.join(words)
print(sentences)
```

```
['A true friend love unconditionally , understands , never judge always try support g
```

Term Frequency and Inverse Document Frequency

```
ps=PorterStemmer()
wordnet=WordNetLemmatizer()
```

```
sentences=nltk.sent_tokenize(paragraph)
corpus=[]
for i in range (len(sentences)):
    review=re.sub('[^a-zA-Z]',' ',sentences[i])
    review=review.lower()
    review=review.split()
    review=[wordnet.lemmatize(word) for word in review if word not in set(stopwords.wor
    review=' '.join(review)
    corpus.append(review)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
cv=TfidfVectorizer()
x=cv.fit_transform(corpus).toarray()
```

```
print(x)
```

```
[[0.30205477 0.         0.2507314  0.         0.         0.
  0.         0.         0.16299352 0.         0.30205477 0.30205477
  0.         0.         0.         0.30205477 0.         0.
  0.30205477 0.         0.2507314  0.         0.         0.
  0.         0.         0.         0.30205477 0.         0.16299352
  0.30205477 0.30205477 0.30205477 0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.34728953 0.         0.69457906 0.         0.
  0.34728953 0.         0.         0.         0.34728953 0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.34728953 0.         0.         0.18740291
  0.         0.         0.         0.         ]
 [0.         0.         0.41710986 0.         0.         0.
  0.         0.         0.27115153 0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.41710986 0.         0.50249001 0.         0.
  0.50249001 0.         0.         0.         0.         0.27115153
  0.         0.         0.         0.         ]
 [0.         0.3151717  0.         0.         0.         0.26161958
  0.3151717  0.         0.         0.         0.         0.
  0.         0.3151717  0.         0.         0.         0.6303434
  0.         0.         0.26161958 0.         0.         0.
  0.         0.         0.         0.         0.26161958 0.
  0.         0.         0.         0.3151717 ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.54975728 0.         0.         0.
  0.         0.         0.50939697 0.         0.         0.
  0.         0.42284323 0.         0.         0.         0.50939697
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.        ]
 [0.         0.         0.         0.40731311 0.         0.33810486
  0.         0.         0.21979258 0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.40731311 0.
  0.         0.         0.         0.         0.67620973 0.21979258
  0.         0.         0.         0.        ]
 [0.         0.         0.         0.         0.62228701 0.
  0.         0.         0.33579589 0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
```