TEXT PREPROCESSING

TOKENIZATION

Splitting text into words, phrases, or other meaningful elements called tokens

In [1]:
```
import nltk
nltk.download('punkt')
```

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

Out[1]: True

punkt usually refers to pre-traied tokenizer model provided by the nltk library. The punkt tokenizer is used to split text into sentences or words based on punctuation marks and other criteria

In [2]:
```
nltk.download('punkt_tab')

from nltk.tokenize import word_tokenize

sentence = "The cats are running quickly because they are very happy."
tokens = word_tokenize(sentence)
print("Tokens :",tokens)
```

```
Tokens : ['The', 'cats', 'are', 'running', 'quickly', 'because', 'they', 'are', 'very', 'happy', '.']
[nltk_data] Downloading package punkt_tab to
[nltk_data]     C:\Users\Acer\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
```

LOWERCASE

Converting all characters to lowercase to ensure uniformity

In [3]:
```
lowercase_tokens = [i.lower() for i in tokens ]
print("Lowercase tokens :",lowercase_tokens)
```

```
Lowercase tokens : ['the', 'cats', 'are', 'running', 'quickly', 'because', 'they', 'are', 'very', 'happy', '.']
```

In [4]:
```
for i in tokens:
    print(i.lower())
```

```
the
cats
are
running
quickly
because
they
are
very
happy
.
```

STOP WORDS REMOVAL

Eliminating common words that usually do not contribute much meaning to the text analysis

In [5]:
```
from nltk.corpus import stopwords
nltk.download("stopwords")
```

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

Out[5]: True

In [6]:
```
stop_words = set(stopwords.words('english'))
stop_words
```

Out[6]:
```
{'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',
'he',
'her',
'here',
'hers',
'herself',
'him',
'himself',
'his',
'how',
'i',
'if',
'in',
'into',
'is',
'isn',
"isn't",
'it',
"it's",
'its',
'itself',
'just',
'll',
'm',
'ma',
'me',
'mightn',
```

```
"mightn't",
'more',
'most',
'mustn',
"mustn't",
'my',
'myself',
'needn',
"needn't",
'no',
'nor',
'not',
'now',
'o',
'of',
'off',
'on',
'once',
'only',
'or',
'other',
'our',
'ours',
'ourselves',
'out',
'over',
'own',
're',
's',
'same',
'shan',
"shan't",
'she',
"she's",
'should',
"should've",
'shouldn',
"shouldn't",
'so',
'some',
'such',
't',
```

```
'than',
'that',
"that'll",
'the',
'their',
'theirs',
'them',
'themselves',
'then',
'there',
'these',
'they',
'this',
'those',
'through',
'to',
'too',
'under',
'until',
'up',
've',
'very',
'was',
'wasn',
"wasn't",
'we',
'were',
'weren',
"weren't",
'what',
'when',
'where',
'which',
'while',
'who',
'whom',
'why',
'will',
'with',
'won',
"won't",
'wouldn',
```

```
 "wouldn't",
 'y',
 'you',
 "you'd",
 "you'll",
 "you're",
 "you've",
 'your',
 'yours',
 'yourself',
 'yourselves'}
```

In [7]:
```
filtered_tokens = [ i for i in lowercase_tokens if i not in stop_words]
filtered_tokens
```

Out[7]:
```
['cats', 'running', 'quickly', 'happy', '.']
```

STEMMING

Reducing words to their base or root form

In [8]:
```
from nltk.stem import PorterStemmer ,WordNetLemmatizer

stemmer = PorterStemmer()
stemmed_tokens = [stemmer.stem(i) for i in filtered_tokens]
print("stemmed Tokens:",stemmed_tokens)
```

```
stemmed Tokens: ['cat', 'run', 'quickli', 'happi', '.']
```

LEMMATIZATION

Reducing words to their base or dictionary form . Unlike stemming , lemmatization considers the context of the word.

In [12]:
```
lemmatizer = WordNetLemmatizer()

nltk.download('wordnet')
nltk.download('omw-1.4')

lemmatized_tokens = [lemmatizer.lemmatize(i) for i in filtered_tokens]
print('Lemmatized Tokens:',lemmatized_tokens)
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Acer\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Acer\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
Lemmatized Tokens: ['cat', 'running', 'quickly', 'happy', '.']
```

- Choose Lemmatization when accuracy is critical, and you want words in their correct dictionary form.
- Choose Stemming when you need faster processing and cat tolerate some inaccuracies in word forms

  Using both on the same data is generally redundant, as they serve the same underlying purpose but through different methods

HOW TO AVOID PUNCTUATIONS

In [13]:
```
import string
string.punctuation
```

Out[13]:
```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

In [14]:
```
tokens_no_punct = [ i for i in lemmatized_tokens if i not in string.punctuation ]
print('Tokens without Punctuation:',tokens_no_punct)
```

```
Tokens without Punctuation: ['cat', 'running', 'quickly', 'happy']
```