Stop Words

In [1]:
```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

nltk.download('stopwords')

sentence = "Stop words are common words that are often considered to be of lit

words = word_tokenize(sentence)

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

filtered_words = [word for word in words if word.lower() not in stop_words]

print(f"Original words: {words}")
print(f"Filtered words: {filtered_words}")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\ASUS\AppData\Roaming\nltk_data...

Original words: ['Stop', 'words', 'are', 'common', 'words', 'that', 'are', 'o
ften', 'considered', 'to', 'be', 'of', 'little', 'value', 'in', 'text', 'anal
ysis']
Filtered words: ['Stop', 'words', 'common', 'words', 'often', 'considered',
'little', 'value', 'text', 'analysis']

[nltk_data]   Unzipping corpora\stopwords.zip.
```

In [6]:
```
def filtering(sentence):
    words = word_tokenize(sentence)

    stopword_list = set(stopwords.words('english'))

    filtered_words = [word for word in words if word.lower() not in stopword_l

    print(f"Original words: {words}\n")
    print(f"Filtered words: {filtered_words}")
```

In [7]:
```
sentence = "The sun was shining brightly in the sky and a gentle breeze was bl

filtering(sentence)
```

```
Original words: ['The', 'sun', 'was', 'shining', 'brightly', 'in', 'the', 'sk
y', 'and', 'a', 'gentle', 'breeze', 'was', 'blowing', 'through', 'the', 'tree
s']

Filtered words: ['sun', 'shining', 'brightly', 'sky', 'gentle', 'breeze', 'bl
owing', 'trees']
```

Part 2

Adding stop words- Add the customer stopwords "NIL" and "JUNK" in spaCy and remove the stopwords in text.

Spacy usually focuses on object oriented stuff whereas NLTK focuses more on strings

In [13]:
```
import spacy
```

In [14]:
```
nlp = spacy.load('en_core_web_sm')
# en is english and sm is small because we are only using it on a small data

nlp.Defaults.stop_words.add("nil") # we are adding these words in the stop wor
nlp.Defaults.stop_words.add("junk")

text = "This is a JUNK sentence that contains NIL information but is useful fo

doc = nlp(text)

filtered_words = [token.text for token in doc if token.text.lower() not in nlp

print(f"Original: {text}")
print(f'Filtered: {" ".join(filtered_words)}')
```

```
Original: This is a JUNK sentence that contains NIL information but is useful
for testing.
Filtered: sentence contains information useful testing .
```

In [ ]:
```

```