In [1]:
```
import nltk
nltk.download("punkt")
nltk.download("stopwords")
nltk.download("wordnet")
nltk.download("averaged_perceptron_tagger")
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
```

Out[1]: True

# Tokenization

In [3]:
```
from nltk import word_tokenize, sent_tokenize
```

In [21]:
```
corpus = "Sachin was the GOAT of the previous generation. Virat is the GOAT of t
```

In [22]:
```
print(word_tokenize(corpus))
print(sent_tokenize(corpus))
```

```
['Sachin', 'was', 'the', 'GOAT', 'of', 'the', 'previous', 'generation', '.', 'Vir
at', 'is', 'the', 'GOAT', 'of', 'this', 'generation', '.', 'Shubman', 'will', 'b
e', 'the', 'GOAT', 'of', 'the', 'next', 'generation']
['Sachin was the GOAT of the previous generation.', 'Virat is the GOAT of this ge
neration.', 'Shubman will be the GOAT of the next generation']
```

# POS tagging

In [23]:
```
from nltk import pos_tag
```

In [24]:
```
tokens = word_tokenize(corpus)
print(pos_tag(tokens))
```

```
[('Sachin', 'NNP'), ('was', 'VBD'), ('the', 'DT'), ('GOAT', 'NNP'), ('of', 'IN'),
('the', 'DT'), ('previous', 'JJ'), ('generation', 'NN'), ('.', '.'), ('Virat', 'N
NP'), ('is', 'VBZ'), ('the', 'DT'), ('GOAT', 'NNP'), ('of', 'IN'), ('this', 'D
T'), ('generation', 'NN'), ('.', '.'), ('Shubman', 'NNP'), ('will', 'MD'), ('be',
'VB'), ('the', 'DT'), ('GOAT', 'NNP'), ('of', 'IN'), ('the', 'DT'), ('next', 'J
J'), ('generation', 'NN')]
```

# Stop word removal

In [25]:
```
from nltk.corpus import stopwords
stop_words = set(stopwords.words("english"))
```

In [26]:
```
tokens = word_tokenize(corpus)
cleaned_tokens = []
for token in tokens:
  if (token not in stop_words):
    cleaned_tokens.append(token)
print(cleaned_tokens)
```

```
['Sachin', 'GOAT', 'previous', 'generation', '.', 'Virat', 'GOAT', 'generation',
'.', 'Shubman', 'GOAT', 'next', 'generation']
```

# Stemming

In [27]:
```
from nltk.stem import PorterStemmer
```

In [28]:
```
stemmer = PorterStemmer()
```

In [29]:
```
stemmed_tokens = []
for token in cleaned_tokens:
  stemmed = stemmer.stem(token)
  stemmed_tokens.append(stemmed)
print(stemmed_tokens)
```

```
['sachin', 'goat', 'previou', 'gener', '.', 'virat', 'goat', 'gener', '.', 'shubm
an', 'goat', 'next', 'gener']
```

# Lemmatization

In [30]:
```
from nltk.stem import WordNetLemmatizer
```

In [31]:
```
lemmatizer = WordNetLemmatizer()
```

In [32]:
```
lemmatized_tokens = []
for token in cleaned_tokens:
  lemmatized = lemmatizer.lemmatize(token)
  lemmatized_tokens.append(lemmatized)
print(lemmatized_tokens)
```

```
['Sachin', 'GOAT', 'previous', 'generation', '.', 'Virat', 'GOAT', 'generation',
'.', 'Shubman', 'GOAT', 'next', 'generation']
```

# TF-IDF

In [33]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

In [34]:
```
corpus = [
    "Sachin was the GOAT of the previous generation",
    "Virat is the GOAT of the this generation",
    "Shubman will be the GOAT of the next generation"
]
```

In [35]:
```
vectorizer = TfidfVectorizer()
```

In [39]:
```
matrix = vectorizer.fit(corpus)
matrix.vocabulary_
```

Out[39]:
```
{'sachin': 7,
 'was': 12,
 'the': 9,
 'goat': 2,
 'of': 5,
 'previous': 6,
 'generation': 1,
 'virat': 11,
 'is': 3,
 'this': 10,
 'shubman': 8,
 'will': 13,
 'be': 0,
 'next': 4}
```

In [41]:
```
tfidf_matrix = vectorizer.transform(corpus)
print(tfidf_matrix)
```

```
  (0, 12)       0.4286758743128819
  (0, 9)        0.5063657539459899
  (0, 7)        0.4286758743128819
  (0, 6)        0.4286758743128819
  (0, 5)        0.25318287697299496
  (0, 2)        0.25318287697299496
  (0, 1)        0.25318287697299496
  (1, 11)       0.4286758743128819
  (1, 10)       0.4286758743128819
  (1, 9)        0.5063657539459899
  (1, 5)        0.25318287697299496
  (1, 3)        0.4286758743128819
  (1, 2)        0.25318287697299496
  (1, 1)        0.25318287697299496
  (2, 13)       0.39400039808922477
  (2, 9)        0.4654059642457353
  (2, 8)        0.39400039808922477
  (2, 5)        0.23270298212286766
  (2, 4)        0.39400039808922477
  (2, 2)        0.23270298212286766
  (2, 1)        0.23270298212286766
  (2, 0)        0.39400039808922477
```

In [42]:
```
print(vectorizer.get_feature_names_out())
```

```
['be' 'generation' 'goat' 'is' 'next' 'of' 'previous' 'sachin' 'shubman'
 'the' 'this' 'virat' 'was' 'will']
```