In [1]:
```
# ## Import Libraries
import nltk
from nltk.tokenize import word_tokenize
from nltk.tag import pos_tag
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import wordnet
from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd
```

In [2]:
```
import nltk
nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger')
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package punkt_tab to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger_eng.zip.
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
```

Out[2]: True

In [3]:
```
sample_document = "The quick brown fox jumps over the lazy dogs. The dogs bark loudl
```

In [4]:
```
# **a) Tokenization**
tokens = word_tokenize(sample_document)
print("Tokens:", tokens)
```

```
Tokens: ['The', 'quick', 'brown', 'fox', 'jumps', 'over', 'the', 'lazy', 'dogs', '.',
'The', 'dogs', 'bark', 'loudly', 'at', 'the', 'playful', 'fox', '.']
```

In [5]:
```
# **b) POS Tagging**
pos_tags = pos_tag(tokens)
print("\nPOS Tags:", pos_tags)
```

```
POS Tags: [('The', 'DT'), ('quick', 'JJ'), ('brown', 'NN'), ('fox', 'NN'), ('jumps',
'VBZ'), ('over', 'IN'), ('the', 'DT'), ('lazy', 'JJ'), ('dogs', 'NNS'), ('.', '.'),
('The', 'DT'), ('dogs', 'NNS'), ('bark', 'VBP'), ('loudly', 'RB'), ('at', 'IN'), ('th
e', 'DT'), ('playful', 'JJ'), ('fox', 'NN'), ('.', '.')]
```

In [6]:
```
# **c) Stop Words Removal**
stop_words = set(stopwords.words('english'))
filtered_tokens = [word.lower() for word in tokens if word.lower() not in stop_words
print("\nTokens after Stop Word Removal:", filtered_tokens)
```

```
Tokens after Stop Word Removal: ['quick', 'brown', 'fox', 'jumps', 'lazy', 'dogs', 'd
ogs', 'bark', 'loudly', 'playful', 'fox']
```

In [7]:
```
# **d) Stemming**
porter_stemmer = PorterStemmer()
stemmed_tokens = [porter_stemmer.stem(word) for word in filtered_tokens]
print("\nStemmed Tokens:", stemmed_tokens)
```

```
Stemmed Tokens: ['quick', 'brown', 'fox', 'jump', 'lazi', 'dog', 'dog', 'bark', 'loud
li', 'play', 'fox']
```

In [8]:
```
# **e) Lemmatization**
wordnet_lemmatizer = WordNetLemmatizer()
```

In [9]:
```
def get_wordnet_pos(tag):
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('N'):
        return wordnet.NOUN
    elif tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN # Default to noun
```

In [10]:
```
lemmatized_tokens = [wordnet_lemmatizer.lemmatize(word.lower(), pos=get_wordnet_pos(
                     for word, tag in pos_tag(filtered_tokens)]
print("\nLemmatized Tokens:", lemmatized_tokens)
```

```
Lemmatized Tokens: ['quick', 'brown', 'fox', 'jump', 'lazy', 'dog', 'dog', 'bark', 'l
oudly', 'playful', 'fox']
```

In [11]:
```
# ## 3. Document Representation: TF-IDF
corpus = [
    "The quick brown fox jumps over the lazy dogs. The dogs bark loudly at the playf
    "A swift black cat leaps across the sleeping mat. The cat meows softly near the
]
```

In [12]:
```
# Initialize TF-IDF Vectorizer
tfidf_vectorizer = TfidfVectorizer()
```

In [13]:
```
# Fit and transform the corpus
tfidf_matrix = tfidf_vectorizer.fit_transform(corpus)
```

In [14]:
```
# Get the feature names (words in the vocabulary)
feature_names = tfidf_vectorizer.get_feature_names_out()
```

In [15]:
```
# Convert the TF-IDF matrix to a DataFrame for better readability
tfidf_df = pd.DataFrame(tfidf_matrix.toarray(), columns=feature_names)
```

In [16]:
```
print("\nTF-IDF Representation of the Corpus:")
print(tfidf_df)
```

```
TF-IDF Representation of the Corpus:
    across        at      bark    black     brown      cat      dogs  \
0  0.00000  0.199602  0.199602  0.00000  0.199602  0.00000  0.399203
1  0.22613  0.000000  0.000000  0.22613  0.000000  0.45226  0.000000

        fox     jumps      lazy  ...     near      over   playful     quick  \
0  0.399203  0.199602  0.199602  ...  0.00000  0.199602  0.199602  0.199602
1  0.000000  0.000000  0.000000  ...  0.22613  0.000000  0.000000  0.000000

       rug  sleeping   softly    swift       the     warm
0  0.00000   0.00000  0.00000  0.00000  0.568073  0.00000
1  0.22613   0.22613  0.22613  0.22613  0.482680  0.22613

[2 rows x 24 columns]
```

In [17]:
```
# To get the TF-IDF representation of our original sample document (the first row):
print("\nTF-IDF Representation of the Sample Document:")
print(tfidf_df.iloc[0])
```

```
TF-IDF Representation of the Sample Document:
across      0.000000
at          0.199602
bark        0.199602
black       0.000000
brown       0.199602
cat         0.000000
dogs        0.399203
fox         0.399203
jumps       0.199602
lazy        0.199602
leaps       0.000000
loudly      0.199602
mat         0.000000
meows       0.000000
near        0.000000
over        0.199602
playful     0.199602
quick       0.199602
rug         0.000000
sleeping    0.000000
softly      0.000000
swift       0.000000
the         0.568073
warm        0.000000
Name: 0, dtype: float64
```