Working with NLTK Package

NItk Package is Text Processing and NLP Based Package, it has lot of Models, corpus and other methods which can help you with NLP Applications.

you all possible things, like tokenisation and vectorisation, Embeddings, stemming and Lemmatisation... it is easy to use because it donot object oriented technique.

[2]:
```
import nltk
```

[9]:
```
nltk.download('punkt')
nltk.download('punkt_tab')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Imart\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to
[nltk_data]     C:\Users\Imart\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt_tab.zip.
```

[9]:
```
True
```

[7]:
```
corpus = ''' Mr. Shahrukh Like the Vadapav of Mumbai. And Mr. Salman Khan Like Chaat of the New Delhi'''
```

[10]:
```
from nltk.tokenize import sent_tokenize
sentences = sent_tokenize(corpus)
```

[11]:
```
print('sentences :',sentences)
```

```
sentences : [' Mr. Shahrukh Like the Vadapav of Mumbai.', 'And Mr. Salman Khan Like Chaat of the New Delhi']
```

[14]:
```
for sentence in sentences:
    print(sentence)
```

```
 Mr. Shahrukh Like the Vadapav of Mumbai.
And Mr. Salman Khan Like Chaat of the New Delhi
```

[12]:
```
from nltk.tokenize import word_tokenize
words = word_tokenize(corpus)
```

[13]:
```
print('words',words)
```

```
words ['Mr.', 'Shahrukh', 'Like', 'the', 'Vadapav', 'of', 'Mumbai', '.', 'And', 'Mr.', 'Salman', 'Khan', 'Like', 'Chaat', 'of', 'the', 'New', 'Delh
i']
```

[17]:
```
lower_words = [word.lower() for word in words]
```

[19]:
```
print('lower case words :',lower_words) #in spacy package we have to type_cast the word tokeniser.
```

```
lower case words : ['mr.', 'shahrukh', 'like', 'the', 'vadapav', 'of', 'mumbai', '.', 'and', 'mr.', 'salman', 'khan', 'like', 'chaat', 'of', 'the',
'new', 'delhi']
```

[20]:
```
vocabs = set(lower_words)
```

[21]:
```
print('vocabs of unique words',vocabs)
```

```
vocabs of unique words {'new', 'like', 'the', 'and', 'mumbai', 'chaat', 'khan', 'delhi', '.', 'salman', 'shahrukh', 'of', 'mr.', 'vadapav'}
```

[22]:
```
# Grammer : Rule of the English.
```

[23]:
```
#if we free the vocabs no body can modify it and hence it can at as grammer rule.
grammer = frozenset(vocabs)
```

[24]:
```
print('Grammer:',grammer)
```

```
Grammer: frozenset({'new', 'like', 'the', 'and', 'mumbai', 'chaat', 'khan', 'delhi', '.', 'salman', 'shahrukh', 'of', 'mr.', 'vadapav'})
```

[26]:
```
print(grammer[1]) # No body can touch it.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[26], line 1
----> 1 print(grammer[1]) # No body can touch it.

TypeError: 'frozenset' object is not subscriptable
```

[ ]: