jupyter Practicle7 Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [1]:
```
!pip install nltk
```
```
Collecting nltk
  Obtaining dependency information for nltk from https://files.pythonhosted.org/packages/a6/0a/0d20d2c0f16be91b9fa32a77b76c60f9
baf6eba419e5ef5deca17af9c582/nltk-3.8.1-py3-none-any.whl.metadata
  Downloading nltk-3.8.1-py3-none-any.whl.metadata (2.8 kB)
Requirement already satisfied: click in c:\users\ghans\appdata\local\programs\python\python311\lib\site-packages (from nltk)
(8.1.3)

Installing collected packages: tqdm, regex, joblib, nltk
Successfully installed joblib-1.4.0 nltk-3.8.1 regex-2023.12.25 tqdm-4.66.2

[notice] A new release of pip is available: 23.2.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

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

In [6]:
```
nltk.download('punkt')
```
```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\ghans\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```
Out[6]:
```
True
```

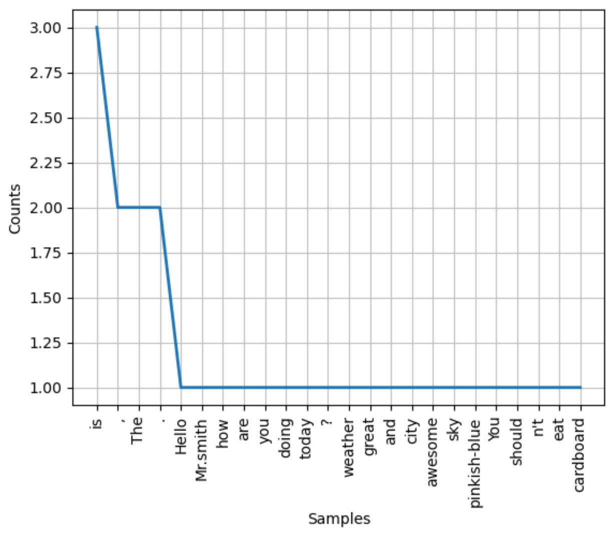
In [7]:
```
from nltk.tokenize import sent_tokenize
text="""Hello Mr.smith,how are you doing today? The weather is great, and city is awesome. The sky is pinkish-blue. You shouldn't
tokenized_text = sent_tokenize(text)
print(tokenized_text)
```
```
['Hello Mr.smith,how are you doing today?', 'The weather is great, and city is awesome.', 'The sky is pinkish-blue.', "You shou
ldn't eat cardboard"]
```

In [8]:
```
from nltk.tokenize import word_tokenize
tokenized_word=word_tokenize(text)
print(tokenized_word)
```
```
['Hello', 'Mr.smith', ',', 'how', 'are', 'you', 'doing', 'today', '?', 'The', 'weather', 'is', 'great', ',', 'and', 'city', 'i
s', 'awesome', '.', 'The', 'sky', 'is', 'pinkish-blue', '.', 'You', 'should', "n't", 'eat', 'cardboard']
```

In [9]:
```
from nltk.probability import FreqDist
fdist = FreqDist(tokenized_word)
print(fdist)
```
```
<FreqDist with 24 samples and 29 outcomes>
```

In [10]:
```
fdist.most_common(2)
```
Out[10]:
```
[('is', 3), (',', 2)]
```

In [11]:
```
import matplotlib.pyplot as plt
fdist.plot(30,cumulative=False)
plt.show()
```



In [12]:
```
nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = set(stopwords.words("english"))
print(stop_words)
```

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

```
{"that'll", 'does', 'shan', 'are', 'has', 'your', 'wasn', "you've", "she's", 'they', 'some', 'mightn', 'over', 'myself', 'own',
'themselves', 'ma', "won't", 'couldn', 't', 'haven', 've', 'i', 'under', 'he', 'only', 'just', 'yours', 'very', 'into', "was
n't", 'yourself', 'again', 'nor', 'now', 'our', 'there', 'when', 'am', 'on', 'to', 'than', "it's", 'how', 'was', 'and', 'in',
'other', 'weren', 'then', 'herself', 'had', 'more', 'whom', 'be', 'didn', "isn't", 'her', 'the', 'through', 'is', 'against', 'a
fter', 'once', 'y', 'hasn', 'wouldn', 'both', 'their', 'most', "mightn't", 'off', 'too', 'those', 'about', 'who', 'doing', 'bu
t', 'which', 's', 'ain', 'this', 'isn', 'out', 'for', 'here', 'we', 'them', 'until', 'what', 'm', 'a', 'why', 'she', "needn't",
'can', 'down', "aren't", 'should', "doesn't", 'd', 'don', "shouldn't", 'himself', 'further', 'by', 'that', 'mustn', "you'll",
'you', 'below', 'up', 'between', 'were', 'above', 'll', 'at', 'each', 'its', "mustn't", "wouldn't", 'where', "you'd", 'itself',
'these', "weren't", "should've", "hadn't", 'as', 'my', 'his', 'being', 'if', 'such', 'aren', 'no', 'from', "don't", 'did', 'hav
ing', 'not', 'will', 're', 'have', 'been', 'him', 'few', "you're", 'because', 'ours', 'before', "haven't", 'during', 'so', 'al
l', "shan't", "couldn't", 'hadn', 'or', 'me', 'an', 'shouldn', 'hers', 'yourselves', 'it', 'same', 'o', 'doesn', 'won', 'with',
'theirs', "didn't", 'while', 'ourselves', "hasn't", 'of', 'needn', 'do', 'any'}
```

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

In [13]:
```
from nltk.tokenize import word_tokenize
text1="""Hello Mr.smith,how are you doing today?"""
tokenized_sent=word_tokenize(text1)
print(tokenized_sent)
filtered_sent=[]
for w in tokenized_sent:
    if w not in stop_words:
        filtered_sent.append(w)
print("Tokenized Sentences:",tokenized_sent)
print("Filtered Sentence:",filtered_sent)
```

```
['Hello', 'Mr.smith', ',', 'how', 'are', 'you', 'doing', 'today', '?']
Tokenized Sentences: ['Hello', 'Mr.smith', ',', 'how', 'are', 'you', 'doing', 'today', '?']
Filtered Sentence: ['Hello', 'Mr.smith', ',', 'today', '?']
```

In [14]:
```
from nltk.stem import PorterStemmer
from nltk.tokenize import sent_tokenize, word_tokenize

ps = PorterStemmer()

stemmed_words=[]
for w in filtered_sent:
    stemmed_words.append(ps.stem(w))

print("Filtered Sentence:",filtered_sent)
print("Stemmed Sentence:",stemmed_words)
```

```
Filtered Sentence: ['Hello', 'Mr.smith', ',', 'today', '?']
Stemmed Sentence: ['hello', 'mr.smith', ',', 'today', '?']
```

In [16]:
```
nltk.download('wordnet')
nltk.download('omw-1.4')
from nltk.stem.wordnet import WordNetLemmatizer
lem = WordNetLemmatizer()

from nltk.stem.porter import PorterStemmer
stem = PorterStemmer()

word = "flying"
print("Lemmenized word:",lem.lemmatize(word,"v"))
print("Stemmed word:",stem.stem(word))
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\ghans\AppData\Roaming\nltk_data...
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\ghans\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

```
Lemmenized word: fly
Stemmed word: fli
```

In [17]:
```
sent = "Albert Einstein was born in Ulm,Germant in 1879."
```

In [18]:
```
tokens=nltk.word_tokenize(sent)
print(tokens)
```

```
['Albert', 'Einstein', 'was', 'born', 'in', 'Ulm', ',', 'Germant', 'in', '1879', '.']
```

In [19]:
```
nltk.download('averaged_perceptron_tagger')
nltk.pos_tag(tokens)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\ghans\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
```

Out[19]:
```
[('Albert', 'NNP'),
 ('Einstein', 'NNP'),
 ('was', 'VBD'),
 ('born', 'VBN'),
 ('in', 'IN'),
 ('Ulm', 'NNP'),
 (',', ','),
 ('Germant', 'NNP'),
 ('in', 'IN'),
 ('1879', 'CD'),
 ('.', '.')]
```

In [20]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
In [21]: 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"
         ]
         vectorizer = TfidfVectorizer()
```

```
In [22]: matrix = vectorizer.fit(corpus)
         matrix.vocabulary_
```

```
Out[22]: {'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 [23]: 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 [24]: print(vectorizer.get_feature_names_out())
```

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

```
In [ ]:
```