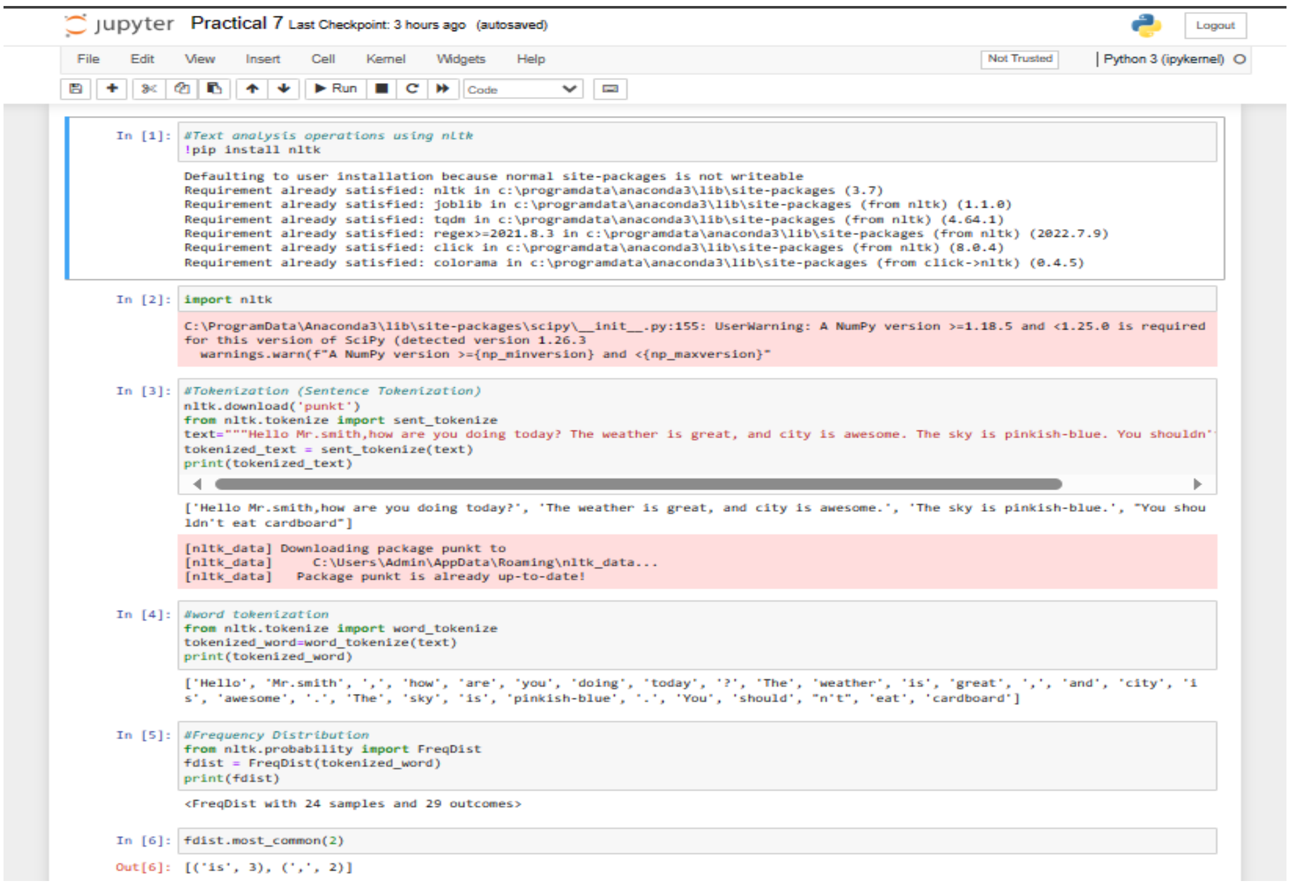
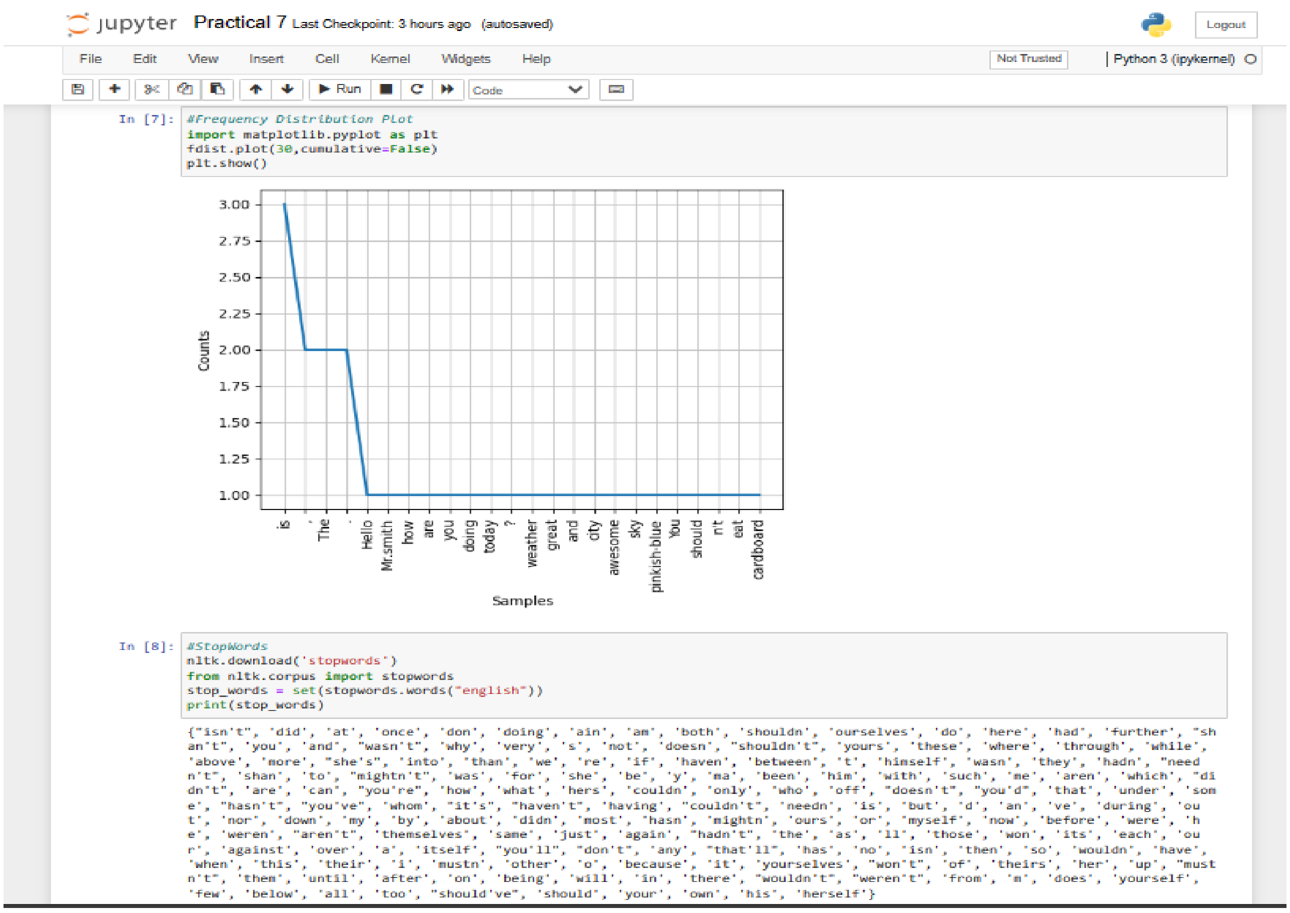
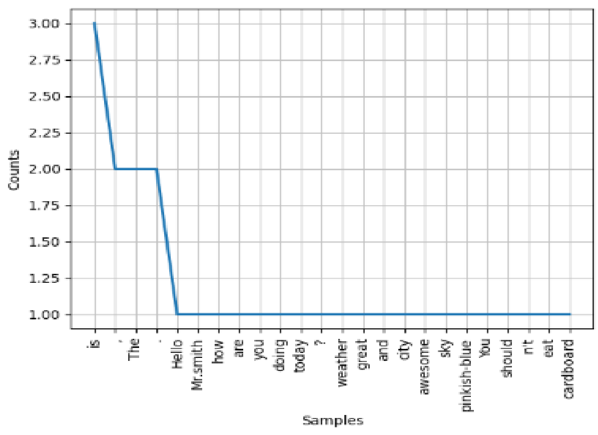

jupyter Practical 7 Last Checkpoint: 3 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

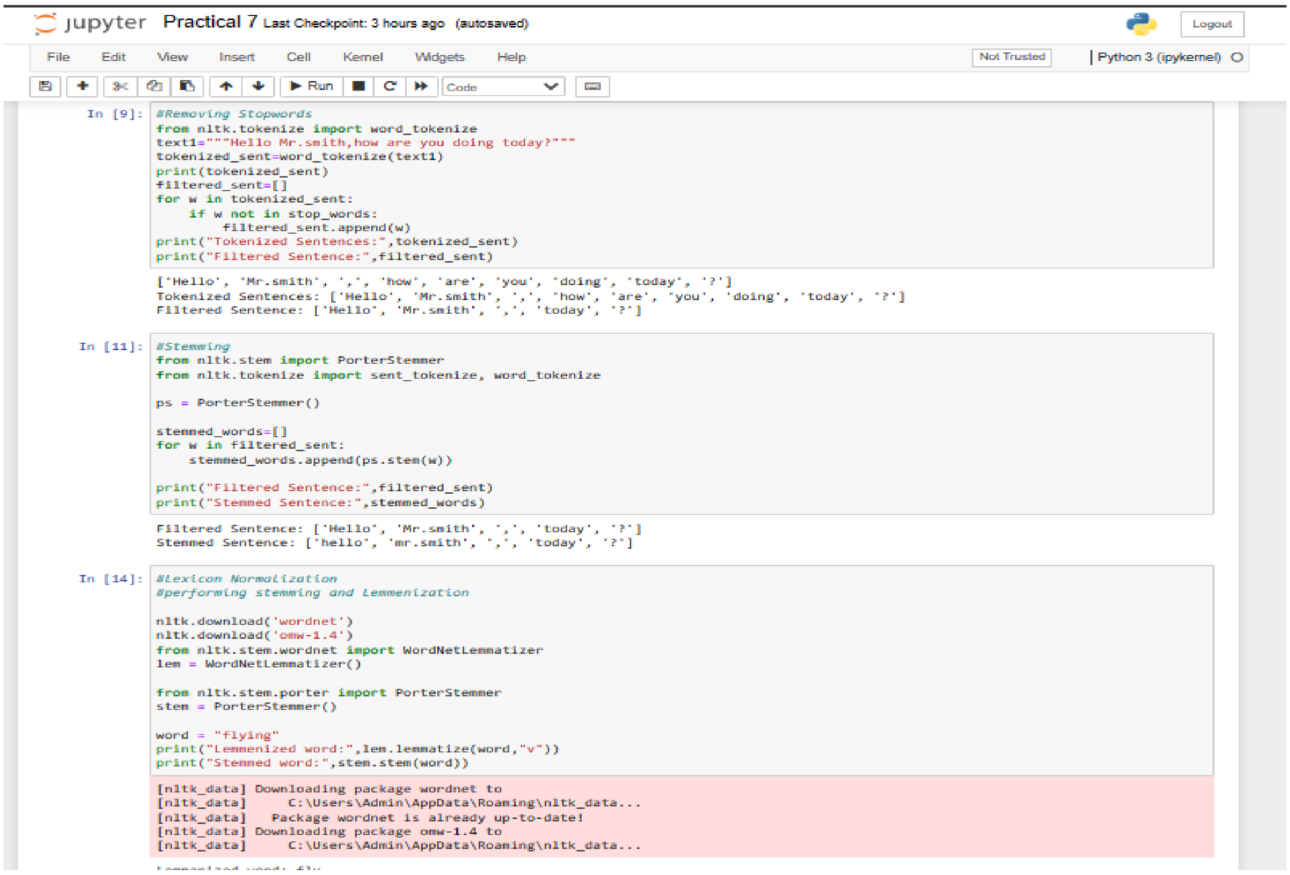
```
In [1]: #Text analysis operations using nltk
        !pip install nltk

        Defaulting to user installation because normal site-packages is not writeable
        Requirement already satisfied: nltk in c:\programdata\anaconda3\lib\site-packages (3.7)
        Requirement already satisfied: joblib in c:\programdata\anaconda3\lib\site-packages (from nltk) (1.1.0)
        Requirement already satisfied: tqdm in c:\programdata\anaconda3\lib\site-packages (from nltk) (4.64.1)
        Requirement already satisfied: regex>=2021.8.3 in c:\programdata\anaconda3\lib\site-packages (from nltk) (2022.7.9)
        Requirement already satisfied: click in c:\programdata\anaconda3\lib\site-packages (from nltk) (8.0.4)
        Requirement already satisfied: colorama in c:\programdata\anaconda3\lib\site-packages (from click->nltk) (0.4.5)

In [2]: import nltk

        C:\ProgramData\Anaconda3\lib\site-packages\scipy\__init__.py:155: UserWarning: A NumPy version >=1.18.5 and <1.25.0 is required
        for this version of SciPy (detected version 1.26.3
          warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"

In [3]: #Tokenization (Sentence Tokenization)
        nltk.download('punkt')
        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'
        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"]

        [nltk_data] Downloading package punkt to
        [nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
        [nltk_data]   Package punkt is already up-to-date!

In [4]: #word tokenization
        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 [5]: #Frequency Distribution
        from nltk.probability import FreqDist
        fdist = FreqDist(tokenized_word)
        print(fdist)

        <FreqDist with 24 samples and 29 outcomes>

In [6]: fdist.most_common(2)

Out[6]: [('is', 3), (',', 2)]
```


jupyter Practical 7 Last Checkpoint: 3 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

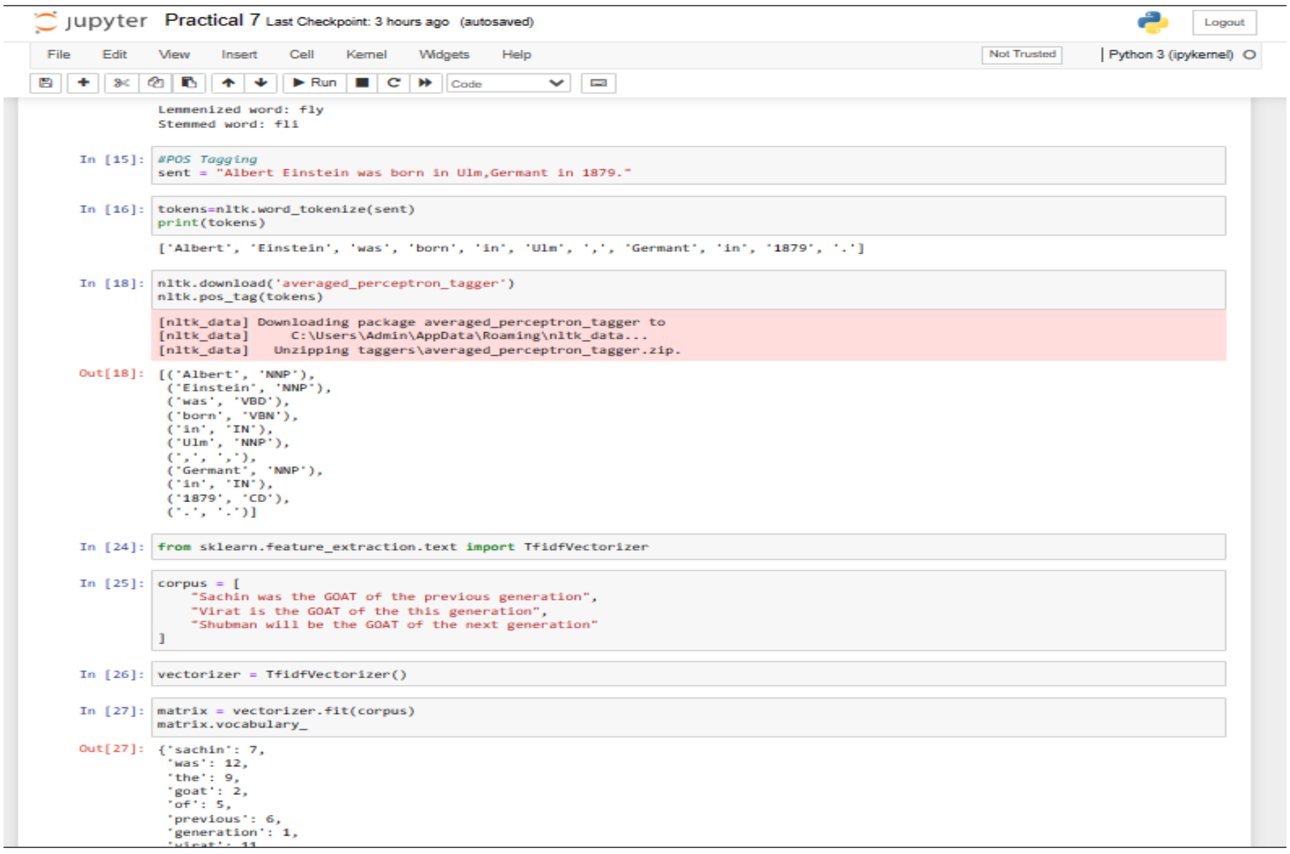
```
In [7]: #Frequency Distribution Plot
        import matplotlib.pyplot as plt
        fdist.plot(30,cumulative=False)
        plt.show()
```



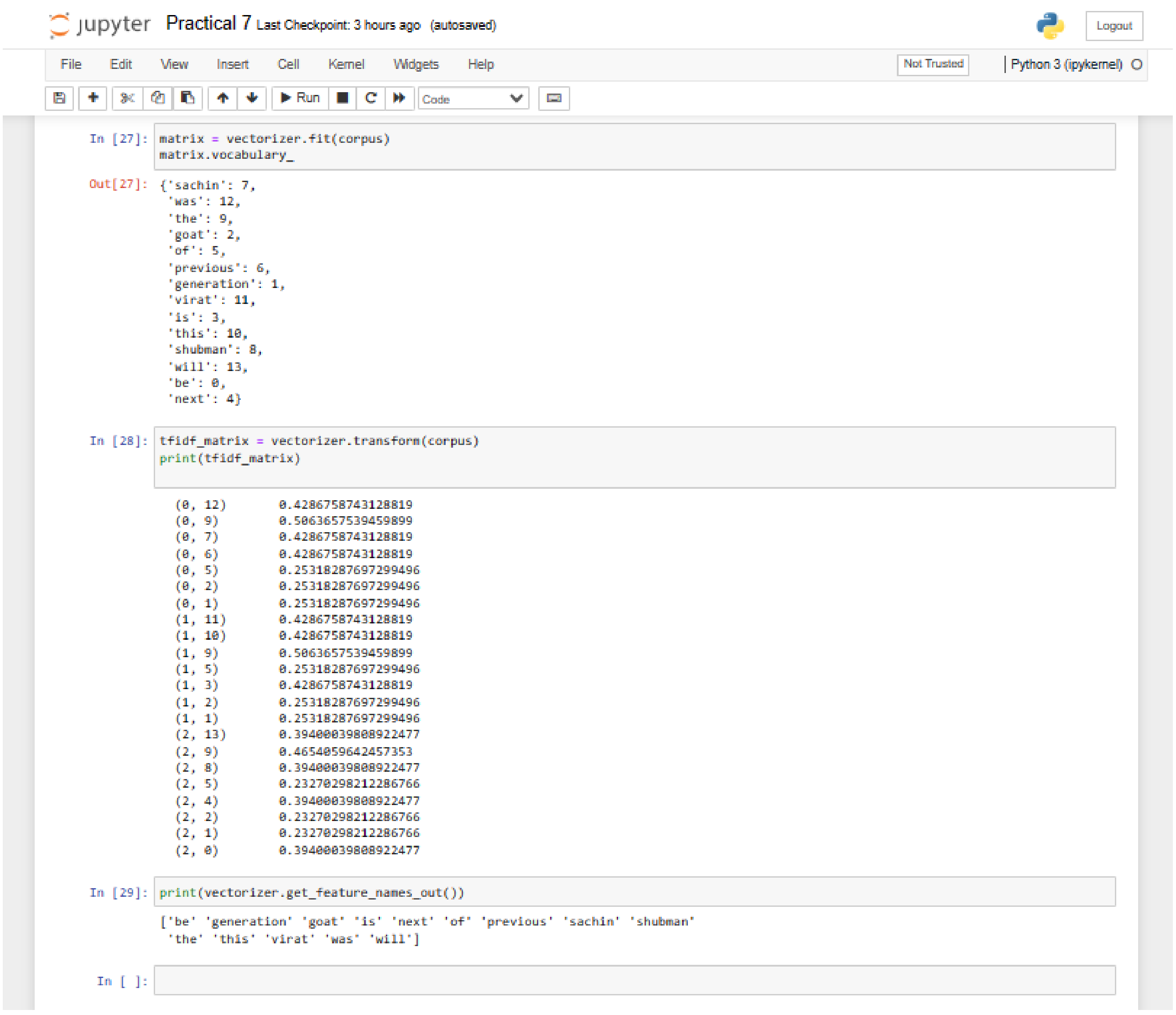
```
In [8]: #StopWords
        nltk.download('stopwords')
        from nltk.corpus import stopwords
        stop_words = set(stopwords.words("english"))
        print(stop_words)

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



```
In [9]:  #Removing Stopwords
         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 [11]: #Stemming
         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 [14]: #Lexicon Normalization
         #performing stemming and Lemmenization

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



```
         Lemmenized word: fly
         Stemmed word: fli

In [15]: #POS Tagging
         sent = "Albert Einstein was born in Ulm,Germant in 1879."

In [16]: tokens=nltk.word_tokenize(sent)
         print(tokens)

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

In [18]: nltk.download('averaged_perceptron_tagger')
         nltk.pos_tag(tokens)

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

Out[18]: [('Albert', 'NNP'),
          ('Einstein', 'NNP'),
          ('was', 'VBD'),
          ('born', 'VBN'),
          ('in', 'IN'),
          ('Ulm', 'NNP'),
          (',', ','),
          ('Germant', 'NNP'),
          ('in', 'IN'),
          ('1879', 'CD'),
          ('.', '.')]

In [24]: from sklearn.feature_extraction.text import TfidfVectorizer

In [25]: 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 [26]: vectorizer = TfidfVectorizer()

In [27]: matrix = vectorizer.fit(corpus)
         matrix.vocabulary_

Out[27]: {'sachin': 7,
          'was': 12,
          'the': 9,
          'goat': 2,
          'of': 5,
          'previous': 6,
          'generation': 1,
```



In [27]:
```
matrix = vectorizer.fit(corpus)
matrix.vocabulary_
```

Out[27]:
```
{'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 [28]:
```
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 [29]:
```
print(vectorizer.get_feature_names_out())
```

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

In [ ]: