In [2]:
```
import os
import nltk
#nltk.download()
```

In [237…
```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
import nltk.corpus
```

In [16]:
```
AI = '''AI, machine learning and deep learning are common terms in enterprise
                IT and sometimes used interchangeably, especially by companies i
                But there are distinctions. The term AI, coined in the 1950s, re
                intelligence by machines. It covers an ever-changing set of capa
                are developed. Technologies that come under the umbrella of AI i
                deep learning. Machine learning enables software applications to
                predicting outcomes without being explicitly programmed to do so
                use historical data as input to predict new output values. This
                effective with the rise of large data sets to train on. Deep lea
                learning, is based on our understanding of how the brain is stru
                use of artificial neural networks structure is the underpinning
                including self-driving cars and ChatGPT.'''
```

In [18]:
```
type(AI)
```

Out[18]:
```
str
```

In [20]:
```
AI
```

Out[20]:
```
"AI, machine learning and deep learning are common terms in enterprise \n
IT and sometimes used interchangeably, especially by companies in their marketi
ng materials. \n                But there are distinctions. The term AI, coined
in the 1950s, refers to the simulation of human \n                intelligence
by machines. It covers an ever-changing set of capabilities as new technologies
\n                are developed. Technologies that come under the umbrella of A
I include machine learning and \n                deep learning. Machine learnin
g enables software applications to become more accurate at \n                pr
edicting outcomes without being explicitly programmed to do so. Machine learnin
g algorithms \n                use historical data as input to predict new outp
ut values. This approach became vastly more \n                effective with th
e rise of large data sets to train on. Deep learning, a subset of machine \n
learning, is based on our understanding of how the brain is structured. Deep le
arning's \n                use of artificial neural networks structure is the u
nderpinning of recent advances in AI, \n                including self-driving
cars and ChatGPT."
```

In [22]:
```
from nltk.tokenize import word_tokenize
```

In [52]:
```
AI_tokens=word_tokenize(AI)
AI_tokens
```

Out[52]:
```
['AI',
 ',',
 'machine',
 'learning',
 'and',
 'deep',
 'learning',
 'are',
 'common',
 'terms',
 'in',
 'enterprise',
 'IT',
 'and',
 'sometimes',
 'used',
 'interchangeably',
 ',',
 'especially',
 'by',
 'companies',
 'in',
 'their',
 'marketing',
 'materials',
 '.',
 'But',
 'there',
 'are',
 'distinctions',
 '.',
 'The',
 'term',
 'AI',
 ',',
 'coined',
 'in',
 'the',
 '1950s',
 ',',
 'refers',
 'to',
 'the',
 'simulation',
 'of',
 'human',
 'intelligence',
 'by',
 'machines',
 '.',
 'It',
 'covers',
 'an',
 'ever-changing',
 'set',
 'of',
 'capabilities',
 'as',
 'new',
 'technologies',
```

```
'are',
'developed',
'.',
'Technologies',
'that',
'come',
'under',
'the',
'umbrella',
'of',
'AI',
'include',
'machine',
'learning',
'and',
'deep',
'learning',
'.',
'Machine',
'learning',
'enables',
'software',
'applications',
'to',
'become',
'more',
'accurate',
'at',
'predicting',
'outcomes',
'without',
'being',
'explicitly',
'programmed',
'to',
'do',
'so',
'.',
'Machine',
'learning',
'algorithms',
'use',
'historical',
'data',
'as',
'input',
'to',
'predict',
'new',
'output',
'values',
'.',
'This',
'approach',
'became',
'vastly',
'more',
'effective',
'with',
'the',
```

```
 'rise',
 'of',
 'large',
 'data',
 'sets',
 'to',
 'train',
 'on',
 '.',
 'Deep',
 'learning',
 ',',
 'a',
 'subset',
 'of',
 'machine',
 'learning',
 ',',
 'is',
 'based',
 'on',
 'our',
 'understanding',
 'of',
 'how',
 'the',
 'brain',
 'is',
 'structured',
 '.',
 'Deep',
 'learning',
 "'s",
 'use',
 'of',
 'artificial',
 'neural',
 'networks',
 'structure',
 'is',
 'the',
 'underpinning',
 'of',
 'recent',
 'advances',
 'in',
 'AI',
 ',',
 'including',
 'self-driving',
 'cars',
 'and',
 'ChatGPT',
 '.']
```

In [54]:
```
len(AI_tokens)
```

Out[54]: 174

In [28]:
```
len(AI_tokens)
```

Out[28]:
```
174
```

In [30]:
```
from nltk.tokenize import sent_tokenize
```

In [32]:
```
AI_tokens=sent_tokenize(AI)
AI_tokens
```

Out[32]:
```
['AI, machine learning and deep learning are common terms in enterprise \n
IT and sometimes used interchangeably, especially by companies in their marketi
ng materials.',
 'But there are distinctions.',
 'The term AI, coined in the 1950s, refers to the simulation of human \n
intelligence by machines.',
 'It covers an ever-changing set of capabilities as new technologies \n
are developed.',
 'Technologies that come under the umbrella of AI include machine learning and
\n                deep learning.',
 'Machine learning enables software applications to become more accurate at \n
predicting outcomes without being explicitly programmed to do so.',
 'Machine learning algorithms \n                use historical data as input to
predict new output values.',
 'This approach became vastly more \n                effective with the rise of
large data sets to train on.',
 'Deep learning, a subset of machine \n                learning, is based on ou
r understanding of how the brain is structured.',
 "Deep learning's \n                use of artificial neural networks structure
is the underpinning of recent advances in AI, \n                including self-
driving cars and ChatGPT."]
```

In [34]:
```
from nltk.tokenize import blankline_tokenize #paragraph
```

In [36]:
```
AI_tokens=blankline_tokenize(AI)
AI_tokens
```

Out[36]:
```
["AI, machine learning and deep learning are common terms in enterprise \n
IT and sometimes used interchangeably, especially by companies in their marketi
ng materials. \n                But there are distinctions. The term AI, coined
in the 1950s, refers to the simulation of human \n                intelligence
by machines. It covers an ever-changing set of capabilities as new technologies
\n                are developed. Technologies that come under the umbrella of A
I include machine learning and \n                deep learning. Machine learnin
g enables software applications to become more accurate at \n                pr
edicting outcomes without being explicitly programmed to do so. Machine learnin
g algorithms \n                use historical data as input to predict new outp
ut values. This approach became vastly more \n                effective with th
e rise of large data sets to train on. Deep learning, a subset of machine \n
learning, is based on our understanding of how the brain is structured. Deep le
arning's \n                use of artificial neural networks structure is the u
nderpinning of recent advances in AI, \n                including self-driving
cars and ChatGPT."]
```

In [38]:
```
len(AI_tokens)
```

Out[38]:
```
1
```

In [42]:
```
from nltk.tokenize import WhitespaceTokenizer
```

In [48]:
```
wt=WhitespaceTokenizer().tokenize(AI)
wt
```

Out[48]:
```
['AI,',
 'machine',
 'learning',
 'and',
 'deep',
 'learning',
 'are',
 'common',
 'terms',
 'in',
 'enterprise',
 'IT',
 'and',
 'sometimes',
 'used',
 'interchangeably,',
 'especially',
 'by',
 'companies',
 'in',
 'their',
 'marketing',
 'materials.',
 'But',
 'there',
 'are',
 'distinctions.',
 'The',
 'term',
 'AI,',
 'coined',
 'in',
 'the',
 '1950s,',
 'refers',
 'to',
 'the',
 'simulation',
 'of',
 'human',
 'intelligence',
 'by',
 'machines.',
 'It',
 'covers',
 'an',
 'ever-changing',
 'set',
 'of',
 'capabilities',
 'as',
 'new',
 'technologies',
 'are',
 'developed.',
 'Technologies',
 'that',
 'come',
 'under',
 'the',
```

```
'umbrella',
'of',
'AI',
'include',
'machine',
'learning',
'and',
'deep',
'learning.',
'Machine',
'learning',
'enables',
'software',
'applications',
'to',
'become',
'more',
'accurate',
'at',
'predicting',
'outcomes',
'without',
'being',
'explicitly',
'programmed',
'to',
'do',
'so.',
'Machine',
'learning',
'algorithms',
'use',
'historical',
'data',
'as',
'input',
'to',
'predict',
'new',
'output',
'values.',
'This',
'approach',
'became',
'vastly',
'more',
'effective',
'with',
'the',
'rise',
'of',
'large',
'data',
'sets',
'to',
'train',
'on.',
'Deep',
'learning,',
'a',
```

```
 'subset',
 'of',
 'machine',
 'learning,',
 'is',
 'based',
 'on',
 'our',
 'understanding',
 'of',
 'how',
 'the',
 'brain',
 'is',
 'structured.',
 'Deep',
 "learning's",
 'use',
 'of',
 'artificial',
 'neural',
 'networks',
 'structure',
 'is',
 'the',
 'underpinning',
 'of',
 'recent',
 'advances',
 'in',
 'AI,',
 'including',
 'self-driving',
 'cars',
 'and',
 'ChatGPT.']
```

In [50]:
```
len(wt)
```

Out[50]: 156

In [56]:
```
len(AI_tokens)
```

Out[56]: 174

In [58]:
```
s='good apple cost $3.88 in hyderabad.please buy two of them. Thanks'
```

In [60]:
```
s
```

Out[60]: 'good apple cost $3.88 in hyderabad.please buy two of them. Thanks'

In [62]:
```
from nltk.tokenize import wordpunct_tokenize
```

In [70]:
```
wp=wordpunct_tokenize(s)
wp
```

Out[70]:
```
['good',
 'apple',
 'cost',
 '$',
 '3',
 '.',
 '88',
 'in',
 'hyderabad',
 '.',
 'please',
 'buy',
 'two',
 'of',
 'them',
 '.',
 'Thanks']
```

In [72]:
```
len(wp)
```

Out[72]:
```
17
```

In [76]:
```
from nltk.util import bigrams,trigrams, ngrams
```

In [78]:
```
string='hello the best and most beautiful thing cannot be seen or touched,it sho
```

In [80]:
```
string
```

Out[80]:
```
'hello the best and most beautiful thing cannot be seen or touched,it should be
felt'
```

In [84]:
```
str_tokens=nltk.word_tokenize(string)
str_tokens
```

Out[84]:
```
['hello',
 'the',
 'best',
 'and',
 'most',
 'beautiful',
 'thing',
 'can',
 'not',
 'be',
 'seen',
 'or',
 'touched',
 ',',
 'it',
 'should',
 'be',
 'felt']
```

In [86]:
```
len(str_tokens)
```

Out[86]:
```
18
```

In [94]:
```
str_bigrams=list(nltk.bigrams(str_tokens))
```

```
str_bigrams
```

Out[94]:
```
[('hello', 'the'),
 ('the', 'best'),
 ('best', 'and'),
 ('and', 'most'),
 ('most', 'beautiful'),
 ('beautiful', 'thing'),
 ('thing', 'can'),
 ('can', 'not'),
 ('not', 'be'),
 ('be', 'seen'),
 ('seen', 'or'),
 ('or', 'touched'),
 ('touched', ','),
 (',', 'it'),
 ('it', 'should'),
 ('should', 'be'),
 ('be', 'felt')]
```

In [98]:
```
str_trigrams=list(nltk.trigrams(str_tokens))
str_trigrams
```

Out[98]:
```
[('hello', 'the', 'best'),
 ('the', 'best', 'and'),
 ('best', 'and', 'most'),
 ('and', 'most', 'beautiful'),
 ('most', 'beautiful', 'thing'),
 ('beautiful', 'thing', 'can'),
 ('thing', 'can', 'not'),
 ('can', 'not', 'be'),
 ('not', 'be', 'seen'),
 ('be', 'seen', 'or'),
 ('seen', 'or', 'touched'),
 ('or', 'touched', ','),
 ('touched', ',', 'it'),
 (',', 'it', 'should'),
 ('it', 'should', 'be'),
 ('should', 'be', 'felt')]
```

In [106…
```
str_ngrams=list(nltk.ngrams(str_tokens,5))
str_ngrams
```

Out[106…
```
[('hello', 'the', 'best', 'and', 'most'),
 ('the', 'best', 'and', 'most', 'beautiful'),
 ('best', 'and', 'most', 'beautiful', 'thing'),
 ('and', 'most', 'beautiful', 'thing', 'can'),
 ('most', 'beautiful', 'thing', 'can', 'not'),
 ('beautiful', 'thing', 'can', 'not', 'be'),
 ('thing', 'can', 'not', 'be', 'seen'),
 ('can', 'not', 'be', 'seen', 'or'),
 ('not', 'be', 'seen', 'or', 'touched'),
 ('be', 'seen', 'or', 'touched', ','),
 ('seen', 'or', 'touched', ',', 'it'),
 ('or', 'touched', ',', 'it', 'should'),
 ('touched', ',', 'it', 'should', 'be'),
 (',', 'it', 'should', 'be', 'felt')]
```

In [114…
```
from nltk.stem import PorterStemmer
```

In [118…
```
pst=PorterStemmer()
```

In [120…
```
pst.stem('Affection')
```

Out[120… 'affect'

In [122…
```
pst.stem('behavioural')
```

Out[122… 'behaviour'

In [124…
```
pst.stem('playing')
```

Out[124… 'play'

In [126…
```
words_to_stem=['give','given','giving','gave','thinking','maximum']
for words in words_to_stem:
    print(words+':'+pst.stem(words))
```

```
give:give
given:given
giving:give
gave:gave
thinking:think
maximum:maximum
```

In [128…
```
from nltk.stem import LancasterStemmer
lst=LancasterStemmer()
```

In [130…
```
words_to_stem=['give','given','giving','gave','thinking','maximum']
for words in words_to_stem:
    print(words+':'+lst.stem(words))
```

```
give:giv
given:giv
giving:giv
gave:gav
thinking:think
maximum:maxim
```

In [134…
```
from nltk.stem import SnowballStemmer
sst=SnowballStemmer('english')
```

In [136…
```
words_to_stem=['give','given','giving','gave','thinking','maximum']
for words in words_to_stem:
    print(words+':'+sst.stem(words))
```

```
give:give
given:given
giving:give
gave:gave
thinking:think
maximum:maximum
```

In [150…
```
from nltk.stem import wordnet
from nltk.stem import WordNetLemmatizer
word_lem=WordNetLemmatizer()
```

In [154…
```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to C:\Users\S
[nltk_data]     SHYAMILI\AppData\Roaming\nltk_data...
```

Out[154… True

In [156…
```
words_to_stem=['give','given','giving','gave','thinking','maximum']
for words in words_to_stem:
    print(words+':'+word_lem.lemmatize(words))
```

```
give:give
given:given
giving:giving
gave:gave
thinking:thinking
maximum:maximum
```

In [158…
```
from nltk.corpus import stopwords
```

In [160…
```
stopwords.words('english')
```

Out[160…

```
['a',
 'about',
 'above',
 'after',
 'again',
 'against',
 'ain',
 'all',
 'am',
 'an',
 'and',
 'any',
 'are',
 'aren',
 "aren't",
 'as',
 'at',
 'be',
 'because',
 'been',
 'before',
 'being',
 'below',
 'between',
 'both',
 'but',
 'by',
 'can',
 'couldn',
 "couldn't",
 'd',
 'did',
 'didn',
 "didn't",
 'do',
 'does',
 'doesn',
 "doesn't",
 'doing',
 'don',
 "don't",
 'down',
 'during',
 'each',
 'few',
 'for',
 'from',
 'further',
 'had',
 'hadn',
 "hadn't",
 'has',
 'hasn',
 "hasn't",
 'have',
 'haven',
 "haven't",
 'having',
 'he',
 "he'd",
```

```
"he'll",
'her',
'here',
'hers',
'herself',
"he's",
'him',
'himself',
'his',
'how',
'i',
"i'd",
'if',
"i'll",
"i'm",
'in',
'into',
'is',
'isn',
"isn't",
'it',
"it'd",
"it'll",
"it's",
'its',
'itself',
"i've",
'just',
'll',
'm',
'ma',
'me',
'mightn',
"mightn't",
'more',
'most',
'mustn',
"mustn't",
'my',
'myself',
'needn',
"needn't",
'no',
'nor',
'not',
'now',
'o',
'of',
'off',
'on',
'once',
'only',
'or',
'other',
'our',
'ours',
'ourselves',
'out',
'over',
'own',
```

```
're',
's',
'same',
'shan',
"shan't",
'she',
"she'd",
"she'll",
"she's",
'should',
'shouldn',
"shouldn't",
"should've",
'so',
'some',
'such',
't',
'than',
'that',
"that'll",
'the',
'their',
'theirs',
'them',
'themselves',
'then',
'there',
'these',
'they',
"they'd",
"they'll",
"they're",
"they've",
'this',
'those',
'through',
'to',
'too',
'under',
'until',
'up',
've',
'very',
'was',
'wasn',
"wasn't",
'we',
"we'd",
"we'll",
"we're",
'were',
'weren',
"weren't",
"we've",
'what',
'when',
'where',
'which',
'while',
'who',
```

```
'whom',
'why',
'will',
'with',
'won',
"won't",
'wouldn',
"wouldn't",
'y',
'you',
"you'd",
"you'll",
'your',
"you're",
'yours',
'yourself',
'yourselves',
"you've"]
```

In [164…

```
stopwords.words('french')
```

Out[164…

```
['au',
 'aux',
 'avec',
 'ce',
 'ces',
 'dans',
 'de',
 'des',
 'du',
 'elle',
 'en',
 'et',
 'eux',
 'il',
 'ils',
 'je',
 'la',
 'le',
 'les',
 'leur',
 'lui',
 'ma',
 'mais',
 'me',
 'même',
 'mes',
 'moi',
 'mon',
 'ne',
 'nos',
 'notre',
 'nous',
 'on',
 'ou',
 'par',
 'pas',
 'pour',
 'qu',
 'que',
 'qui',
 'sa',
 'se',
 'ses',
 'son',
 'sur',
 'ta',
 'te',
 'tes',
 'toi',
 'ton',
 'tu',
 'un',
 'une',
 'vos',
 'votre',
 'vous',
 'c',
 'd',
 'j',
 'l',
```

```
'à',
'm',
'n',
's',
't',
'y',
'été',
'étée',
'étées',
'étés',
'étant',
'étante',
'étants',
'étantes',
'suis',
'es',
'est',
'sommes',
'êtes',
'sont',
'serai',
'seras',
'sera',
'serons',
'serez',
'seront',
'serais',
'serait',
'serions',
'seriez',
'seraient',
'étais',
'était',
'étions',
'étiez',
'étaient',
'fus',
'fut',
'fûmes',
'fûtes',
'furent',
'sois',
'soit',
'soyons',
'soyez',
'soient',
'fusse',
'fusses',
'fût',
'fussions',
'fussiez',
'fussent',
'ayant',
'ayante',
'ayantes',
'ayants',
'eu',
'eue',
'eues',
'eus',
```

```
 'ai',
 'as',
 'avons',
 'avez',
 'ont',
 'aurai',
 'auras',
 'aura',
 'aurons',
 'aurez',
 'auront',
 'aurais',
 'aurait',
 'aurions',
 'auriez',
 'auraient',
 'avais',
 'avait',
 'avions',
 'aviez',
 'avaient',
 'eut',
 'eûmes',
 'eûtes',
 'eurent',
 'aie',
 'aies',
 'ait',
 'ayons',
 'ayez',
 'aient',
 'eusse',
 'eusses',
 'eût',
 'eussions',
 'eussiez',
 'eussent']
```

In [166…
```
len( stopwords.words('english'))
```

Out[166…
```
198
```

In [168…
```
len( stopwords.words('french'))
```

Out[168…
```
157
```

In [176…
```
sent='sam is natural when it comes to drawings'
sent_tokens=word_tokenize(sent)
sent_tokens
```

Out[176…
```
['sam', 'is', 'natural', 'when', 'it', 'comes', 'to', 'drawings']
```

In [180…
```
nltk.download('averaged_perceptron_tagger_eng')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     C:\Users\S SHYAMILI\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger_eng.zip.
```

Out[180…
```
True
```

In [182…
```
for token in sent_tokens:
    print(nltk.pos_tag([token]))
```

```
[('sam', 'NN')]
[('is', 'VBZ')]
[('natural', 'JJ')]
[('when', 'WRB')]
[('it', 'PRP')]
[('comes', 'VBZ')]
[('to', 'TO')]
[('drawings', 'NNS')]
```

In [190…
```
sent2='john is eating a delicious cake'
sent2_tokens=word_tokenize(sent2)
sent2_tokens
```

Out[190…
```
['john', 'is', 'eating', 'a', 'delicious', 'cake']
```

In [192…
```
for token in sent2_tokens:
    print(nltk.pos_tag([token]))
```

```
[('john', 'NN')]
[('is', 'VBZ')]
[('eating', 'VBG')]
[('a', 'DT')]
[('delicious', 'JJ')]
[('cake', 'NN')]
```

In [203…
```
from nltk import ne_chunk
```

In [205…
```
NE_sent='The US president stays in the WHITEHOUSE'
```

In [213…
```
NE_tokens=word_tokenize(NE_sent)
NE_tokens
```

Out[213…
```
['The', 'US', 'president', 'stays', 'in', 'the', 'WHITEHOUSE']
```

In [219…
```
NE_tags=nltk.pos_tag(NE_tokens)
NE_tags
```

Out[219…
```
[('The', 'DT'),
 ('US', 'NNP'),
 ('president', 'NN'),
 ('stays', 'NNS'),
 ('in', 'IN'),
 ('the', 'DT'),
 ('WHITEHOUSE', 'NNP')]
```

In [254…
```
NE_NER=ne_chunk(NE_tags)
NE_NER
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
File ~\AppData\Roaming\Python\Python312\site-packages\IPython\core\formatters.py:
406, in BaseFormatter.__call__(self, obj)
    404     method = get_real_method(obj, self.print_method)
    405     if method is not None:
--> 406         return method()
    407     return None
    408 else:

File ~\anaconda3\Lib\site-packages\nltk\tree\tree.py:782, in Tree._repr_svg_(sel
f)
    781 def _repr_svg_(self):
--> 782     from svgling import draw_tree
    784     return draw_tree(self)._repr_svg_()

ModuleNotFoundError: No module named 'svgling'
```

Out[254…

```
Tree('S', [('The', 'DT'), Tree('GSP', [('US', 'NNP')]), ('president', 'NN'),
('stays', 'NNS'), ('in', 'IN'), ('the', 'DT'), Tree('ORGANIZATION', [('WHITEHOU
SE', 'NNP')])])
```

In [263…

```
!pip install wordcloud
```

```
Collecting wordcloud
  Downloading wordcloud-1.9.4-cp312-cp312-win_amd64.whl.metadata (3.5 kB)
Requirement already satisfied: numpy>=1.6.1 in c:\users\s shyamili\anaconda3\lib
\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\s shyamili\anaconda3\lib\site-p
ackages (from wordcloud) (10.4.0)
Requirement already satisfied: matplotlib in c:\users\s shyamili\anaconda3\lib\si
te-packages (from wordcloud) (3.9.2)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\s shyamili\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\s shyamili\anaconda3\lib
\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\s shyamili\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\s shyamili\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\s shyamili\appdata\roa
ming\python\python312\site-packages (from matplotlib->wordcloud) (24.2)
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\s shyamili\anaconda3
\lib\site-packages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\s shyamili\appdat
a\roaming\python\python312\site-packages (from matplotlib->wordcloud) (2.9.0.post
0)
Requirement already satisfied: six>=1.5 in c:\users\s shyamili\appdata\roaming\py
thon\python312\site-packages (from python-dateutil>=2.7->matplotlib->wordcloud)
(1.17.0)
Downloading wordcloud-1.9.4-cp312-cp312-win_amd64.whl (301 kB)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.9.4
```

In [267…

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
```

In [271…

```
text="Innovation drives progress as technology evolves rapidly across industries
```

In [273…

```
text
```

Out[273…

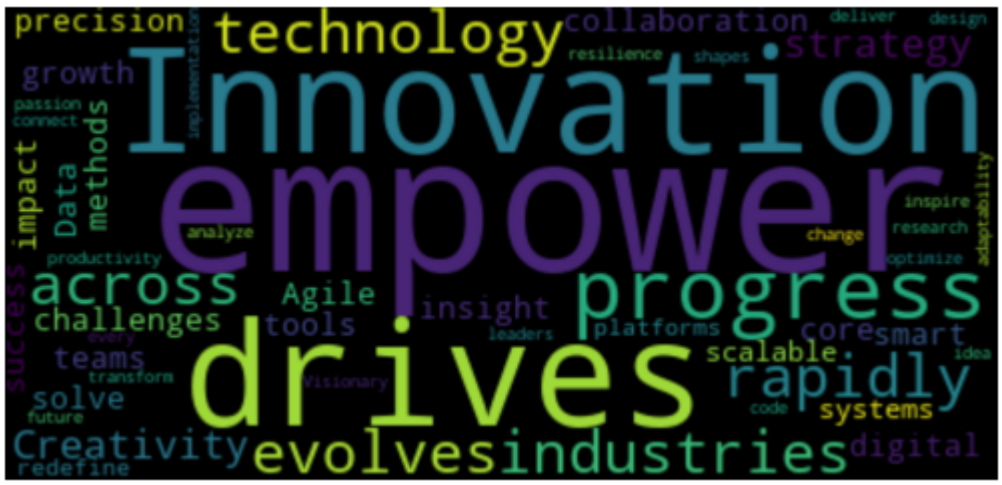
```
'Innovation drives progress as technology evolves rapidly across industries. Cr
eativity, strategy, and collaboration empower teams to solve challenges with pr
ecision. Data, insight, growth, and impact are at the core of success. Agile me
thods, digital tools, smart systems, and scalable platforms redefine productivi
ty. Visionary leaders inspire change through resilience, adaptability, and pass
ion. From code to design, research to implementation, every idea shapes the fut
ure. Empower, transform, analyze, optimize, connect, deliver'
```

In [275…

```
wordcloud=WordCloud(font_path=None,
    width=420,
    height=200,
    margin=2,
    ranks_only=None,
    prefer_horizontal=0.9,
    mask=None,
    scale=1,
    color_func=None,
    max_words=200,
    min_font_size=4,
    stopwords=None,
    random_state=None,
    background_color='black',
    max_font_size=None,
    font_step=1,
    mode='RGB',
    relative_scaling='auto',
    regexp=None,
    collocations=True,
    colormap=None,
    normalize_plurals=True,
    contour_width=0,
    contour_color='black',
    repeat=False,
    include_numbers=False,
    min_word_length=0,
    collocation_threshold=30,).generate(text)
```

In [277…

```
plt.imshow(wordcloud,interpolation='quadric')
plt.axis('off')
plt.margins(x=0,y=0)
plt.show()
```



In [ ]: