Text Preprocessing

Import Required Library

In [3]:

```
import warnings
warnings.filterwarnings("ignore")
import nltk
from nltk import word_tokenize
from nltk.stem import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
nltk.download('stopwords')
import re
import numpy as np
```

```
[nltk_data] Downloading package stopwords to C:\Users\Anjali
[nltk_data]     Sharma\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

We have taken the paragraph

In [4]:

```
paragraph = """Thank you all so very much. Thank you to the Academy. Thank you to all of you in this room.
            I have to congratulate the other incredible nominees this year. The Revenant was the product of the t
ireless efforts of an unbelievable cast and crew.
            First off, to my brother in this endeavor, Mr. Tom Hardy. Tom, your talent on screen can only be surp
assed by your friendship off screen,
            thank you for creating a transcendent cinematic experience. Thank you to everybody at Fox and New Reg
ency my entire team.
            I have to thank everyone from the very onset of my career … To my parents;
            none of this would be possible without you. And to my friends, I love you dearly; you know who you ar
e.

            And lastly, I just want to say this: Making The Revenant was about man's relationship to the natural
world.
            A world that we collectively felt in 2015 as the hottest year in recorded history. Our production nee
ded to move to
            the southern tip of this planet just to be able to find snow.Climate change is real, it is happening
right now.
            It is the most urgent threat facing our entire species, and we need to work collectively together and
stop procrastinating.
            We need to support leaders around the world who do not speak for the big polluters, but who speak for
all of humanity, for the indigenous people of the world,
            for the billions and billions of underprivileged people out there who would be most affected by this.
For our children's children,
            and for those people out there whose voices have been drowned out by the politics of greed. I thank y
ou all for this amazing award tonight.
            Let us not take this planet for granted. I do not take tonight for granted.
            Thank you so very much."""
```

We will use sent_tokenize which return the list of sentences from above paragraph

In [5]:

```
sentence = nltk.sent_tokenize(paragraph) #tokenize the paragraph into sentences
```

In [6]:

```
print(sentence)
```

```
['Thank you all so very much.', 'Thank you to the Academy.', 'Thank you to all of you in this room.'
, 'I have to congratulate the other incredible nominees this year.', 'The Revenant was the product o
f the tireless efforts of an unbelievable cast and crew.', 'First off, to my brother in this endeavo
r, Mr. Tom Hardy.', 'Tom, your talent on screen can only be surpassed by your friendship off screen,
\n            thank you for creating a transcendent cinematic experience.', 'Thank you to everybody
at Fox and New Regency my entire team.', 'I have to thank everyone from the very onset of my career
… To my parents; \n            none of this would be possible without you.', 'And to my friends, I l
ove you dearly; you know who you are.', "And lastly, I just want to say this: Making The Revenant wa
s about man's relationship to the natural world.", 'A world that we collectively felt in 2015 as the
hottest year in recorded history.', 'Our production needed to move to \n            the southern tip
of this planet just to be able to find snow.Climate change is real, it is happening right now.', 'It
is the most urgent threat facing our entire species, and we need to work collectively together and s
top procrastinating.', 'We need to support leaders around the world who do not speak for the big pol
luters, but who speak for all of humanity, for the indigenous people of the world, \n            for
the billions and billions of underprivileged people out there who would be most affected by this.',
'For our children’s children, \n            and for those people out there whose voices have been dr
owned out by the politics of greed.', 'I thank you all for this amazing award tonight.', 'Let us not
take this planet for granted.', 'I do not take tonight for granted.', 'Thank you so very much.']
```

In [7]:

```
print(len(sentence)) #length of sentences
```

```
20
```

Let's use word_tokenize() which tokenize the sentence into words.

In [8]:

```
words = word_tokenize(paragraph) #tokenize the paragraph into word
```

In [9]:

```
print(words)
```

```
['Thank', 'you', 'all', 'so', 'very', 'much', '.', 'Thank', 'you', 'to', 'the', 'Academy', '.', 'Tha
nk', 'you', 'to', 'all', 'of', 'you', 'in', 'this', 'room', '.', 'I', 'have', 'to', 'congratulate',
'the', 'other', 'incredible', 'nominees', 'this', 'year', '.', 'The', 'Revenant', 'was', 'the', 'pro
duct', 'of', 'the', 'tireless', 'efforts', 'of', 'an', 'unbelievable', 'cast', 'and', 'crew', '.', '
First', 'off', ',', 'to', 'my', 'brother', 'in', 'this', 'endeavor', ',', 'Mr.', 'Tom', 'Hardy', '.'
, 'Tom', ',', 'your', 'talent', 'on', 'screen', 'can', 'only', 'be', 'surpassed', 'by', 'your', 'fri
endship', 'off', 'screen', ',', 'thank', 'you', 'for', 'creating', 'a', 'transcendent', 'cinematic',
'experience', '.', 'Thank', 'you', 'to', 'everybody', 'at', 'Fox', 'and', 'New', 'Regency', 'my', 'e
ntire', 'team', '.', 'I', 'have', 'to', 'thank', 'everyone', 'from', 'the', 'very', 'onset', 'of', '
my', 'career', '…', 'To', 'my', 'parents', ';', 'none', 'of', 'this', 'would', 'be', 'possible', 'wi
thout', 'you', '.', 'And', 'to', 'my', 'friends', ',', 'I', 'love', 'you', 'dearly', ';', 'you', 'kn
ow', 'who', 'you', 'are', '.', 'And', 'lastly', ',', 'I', 'just', 'want', 'to', 'say', 'this', ':',
'Making', 'The', 'Revenant', 'was', 'about', 'man', "'s", 'relationship', 'to', 'the', 'natural', 'w
orld', '.', 'A', 'world', 'that', 'we', 'collectively', 'felt', 'in', '2015', 'as', 'the', 'hottest'
, 'year', 'in', 'recorded', 'history', '.', 'Our', 'production', 'needed', 'to', 'move', 'to', 'the'
, 'southern', 'tip', 'of', 'this', 'planet', 'just', 'to', 'be', 'able', 'to', 'find', 'snow.Climate
', 'change', 'is', 'real', ',', 'it', 'is', 'happening', 'right', 'now', '.', 'It', 'is', 'the', 'mo
st', 'urgent', 'threat', 'facing', 'our', 'entire', 'species', ',', 'and', 'we', 'need', 'to', 'work
', 'collectively', 'together', 'and', 'stop', 'procrastinating', '.', 'We', 'need', 'to', 'support',
'leaders', 'around', 'the', 'world', 'who', 'do', 'not', 'speak', 'for', 'the', 'big', 'polluters',
',', 'but', 'who', 'speak', 'for', 'all', 'of', 'humanity', ',', 'for', 'the', 'indigenous', 'people
', 'of', 'the', 'world', ',', 'for', 'the', 'billions', 'and', 'billions', 'of', 'underprivileged',
'people', 'out', 'there', 'who', 'would', 'be', 'most', 'affected', 'by', 'this', '.', 'For', 'our',
'children', '’', 's', 'children', ',', 'and', 'for', 'those', 'people', 'out', 'there', 'whose', 'vo
ices', 'have', 'been', 'drowned', 'out', 'by', 'the', 'politics', 'of', 'greed', '.', 'I', 'thank',
'you', 'all', 'for', 'this', 'amazing', 'award', 'tonight', '.', 'Let', 'us', 'not', 'take', 'this',
'planet', 'for', 'granted', '.', 'I', 'do', 'not', 'take', 'tonight', 'for', 'granted', '.', 'Thank'
, 'you', 'so', 'very', 'much', '.']
```

In [10]:

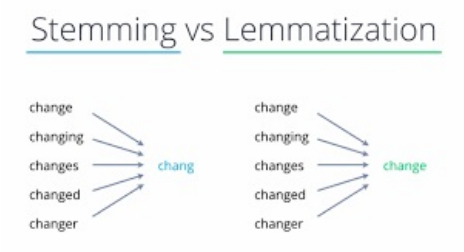
```
print(len(words))
```

```
343
```

Stemming And Lemmatization

Stemming stem the similar words into a word with meaningless .So it takes less time while processing.We can use stemming on that preprocessing analysis where meaning of words are not that much important.

While Lemmatization return meaningful word so it take time while processing. We can use stemming on that preprocessing analysis where meaning of words are important.



Observation

We got "chang" word after stemming but "change" word after Lemmatization.

Applied Stemming on Paragraph

In [11]:

```
stemmer = PorterStemmer()
sentence = nltk.sent_tokenize(paragraph) #tokenize the paragraph into sentences
#stemming

for i in range(len(sentence)): #took the length of sentences
    words = nltk.word_tokenize(sentence[i]) #tokenize the sentences into word
    newwords = [stemmer.stem(word) for word in words]#iterate over the words and found stemming word in newwords
    sentence[i] =" ".join(newwords) # join the all newwords
```

In [12]:

```
print(sentence)
```

```
['thank you all so veri much .', 'thank you to the academi .', 'thank you to all of you in thi room
.', 'I have to congratul the other incred nomine thi year .', 'the reven wa the product of the tirel
ess effort of an unbeliev cast and crew .', 'first off , to my brother in thi endeavor , mr. tom har
di .', 'tom , your talent on screen can onli be surpass by your friendship off screen , thank you fo
r creat a transcend cinemat experi .', 'thank you to everybodi at fox and new regenc my entir team .
', 'I have to thank everyon from the veri onset of my career … To my parent ; none of thi would be p
ossibl without you .', 'and to my friend , I love you dearli ; you know who you are .', "and lastli
, I just want to say thi : make the reven wa about man 's relationship to the natur world .", 'A wor
ld that we collect felt in 2015 as the hottest year in record histori .', 'our product need to move
to the southern tip of thi planet just to be abl to find snow.clim chang is real , it is happen righ
t now .', 'It is the most urgent threat face our entir speci , and we need to work collect togeth an
d stop procrastin .', 'We need to support leader around the world who do not speak for the big pollu
t , but who speak for all of human , for the indigen peopl of the world , for the billion and billio
n of underprivileg peopl out there who would be most affect by thi .', 'for our children ’ s childre
n , and for those peopl out there whose voic have been drown out by the polit of greed .', 'I thank
you all for thi amaz award tonight .', 'let us not take thi planet for grant .', 'I do not take toni
ght for grant .', 'thank you so veri much .']
```

Applied Lemmatizer on Paragraph

In [13]:

```
sentence = nltk.sent_tokenize(paragraph) # tokenize the paragraph into sentences
lemmatizer = WordNetLemmatizer()

#lemmatizing
for i in range(len(sentence)): #took all the sentences
    words = nltk.word_tokenize(sentence[i]) # did word tokenization of each of sentences
    newwords = [lemmatizer.lemmatize(word) for word in words] #lemmatize the word into newwords
    sentence[i]=" ".join(newwords) #join the newwords into sentence list
```

In [14]:

```
print(sentence)
```

```
['Thank you all so very much .', 'Thank you to the Academy .', 'Thank you to all of you in this room
.', 'I have to congratulate the other incredible nominee this year .', 'The Revenant wa the product
of the tireless effort of an unbelievable cast and crew .', 'First off , to my brother in this endea
vor , Mr. Tom Hardy .', 'Tom , your talent on screen can only be surpassed by your friendship off sc
reen , thank you for creating a transcendent cinematic experience .', 'Thank you to everybody at Fox
and New Regency my entire team .', 'I have to thank everyone from the very onset of my career … To m
y parent ; none of this would be possible without you .', 'And to my friend , I love you dearly ; yo
u know who you are .', "And lastly , I just want to say this : Making The Revenant wa about man 's r
elationship to the natural world .", 'A world that we collectively felt in 2015 a the hottest year i
n recorded history .', 'Our production needed to move to the southern tip of this planet just to be
able to find snow.Climate change is real , it is happening right now .', 'It is the most urgent thre
at facing our entire specie , and we need to work collectively together and stop procrastinating .',
'We need to support leader around the world who do not speak for the big polluter , but who speak fo
r all of humanity , for the indigenous people of the world , for the billion and billion of underpri
vileged people out there who would be most affected by this .', 'For our child ’ s child , and for t
hose people out there whose voice have been drowned out by the politics of greed .', 'I thank you al
l for this amazing award tonight .', 'Let u not take this planet for granted .', 'I do not take toni
ght for granted .', 'Thank you so very much .']
```

Stopword Removal

This is a such important step for text preprocessing. It will remove the words like is,am,or,not etc.

In [15]:

```
for i in range(len(sentence)):
    words = nltk.word_tokenize(sentence[i])
    newwords = [word for word in words if word not in stopwords.words('english')]
    sentence[i] = " ".join(newwords)
```

In [16]:

```
print(sentence)
```

```
['Thank much .', 'Thank Academy .', 'Thank room .', 'I congratulate incredible nominee year .', 'The
Revenant wa product tireless effort unbelievable cast crew .', 'First , brother endeavor , Mr. Tom H
ardy .', 'Tom , talent screen surpassed friendship screen , thank creating transcendent cinematic ex
perience .', 'Thank everybody Fox New Regency entire team .', 'I thank everyone onset career … To pa
rent ; none would possible without .', 'And friend , I love dearly ; know .', "And lastly , I want s
ay : Making The Revenant wa man 's relationship natural world .", 'A world collectively felt 2015 ho
ttest year recorded history .', 'Our production needed move southern tip planet able find snow.Clima
te change real , happening right .', 'It urgent threat facing entire specie , need work collectively
together stop procrastinating .', 'We need support leader around world speak big polluter , speak hu
manity , indigenous people world , billion billion underprivileged people would affected .', 'For ch
ild ’ child , people whose voice drowned politics greed .', 'I thank amazing award tonight .', 'Let
u take planet granted .', 'I take tonight granted .', 'Thank much .']
```

How to find part of speech for different-2 word in above sentence

In [17]:

```
words = word_tokenize(paragraph)
tagged_word = nltk.pos_tag(words)
word_tag = []
for i in tagged_word:
    word_tag.append(i[0]+"_"+i[1])
tagged_paragraph = " ".join(word_tag)
```

In [18]:

```
print(tagged_paragraph)
```

```
Thank_NNP you_PRP all_DT so_RB very_RB much_JJ ._. Thank_VB you_PRP to_TO the_DT Academy_NNP ._. Tha
nk_NNP you_PRP to_TO all_DT of_IN you_PRP in_IN this_DT room_NN ._. I_PRP have_VBP to_TO congratulat
e_VB the_DT other_JJ incredible_JJ nominees_NNS this_DT year_NN ._. The_DT Revenant_NNP was_VBD the_
DT product_NN of_IN the_DT tireless_NN efforts_NNS of_IN an_DT unbelievable_JJ cast_NN and_CC crew_N
N ._. First_NNP off_RB ,_, to_TO my_PRP$ brother_NN in_IN this_DT endeavor_NN ,_, Mr._NNP Tom_NNP Ha
rdy_NNP ._. Tom_NNP ,_, your_PRP$ talent_NN on_IN screen_NN can_MD only_RB be_VB surpassed_VBN by_IN
your_PRP$ friendship_NN off_IN screen_NN ,_, thank_NN you_PRP for_IN creating_VBG a_DT transcendent_
JJ cinematic_JJ experience_NN ._. Thank_NNP you_PRP to_TO everybody_VB at_IN Fox_NNP and_CC New_NNP
Regency_NNP my_PRP$ entire_JJ team_NN ._. I_PRP have_VBP to_TO thank_VB everyone_NN from_IN the_DT v
ery_RB onset_NN of_IN my_PRP$ career_NN …_NN To_TO my_PRP$ parents_NNS ;_: none_NN of_IN this_DT wou
ld_MD be_VB possible_JJ without_IN you_PRP ._. And_CC to_TO my_PRP$ friends_NNS ,_, I_PRP love_VBP y
ou_PRP dearly_RB ;_: you_PRP know_VBP who_WP you_PRP are_VBP ._. And_CC lastly_RB ,_, I_PRP just_RB
want_VBP to_TO say_VB this_DT :_: Making_VBG The_DT Revenant_NNP was_VBD about_IN man_NN 's_POS rela
tionship_NN to_TO the_DT natural_JJ world_NN ._. A_DT world_NN that_IN we_PRP collectively_RB felt_V
BD in_IN 2015_CD as_IN the_DT hottest_JJS year_NN in_IN recorded_JJ history_NN ._. Our_PRP$ producti
on_NN needed_VBN to_TO move_VB to_TO the_DT southern_JJ tip_NN of_IN this_DT planet_NN just_RB to_TO
be_VB able_JJ to_TO find_VB snow.Climate_JJ change_NN is_VBZ real_JJ ,_, it_PRP is_VBZ happening_VBG
right_RB now_RB ._. It_PRP is_VBZ the_DT most_RBS urgent_JJ threat_NN facing_VBG our_PRP$ entire_JJ
species_NNS ,_, and_CC we_PRP need_VBP to_TO work_VB collectively_RB together_RB and_CC stop_VB proc
rastinating_NN ._. We_PRP need_VBP to_TO support_VB leaders_NNS around_IN the_DT world_NN who_WP do_
VBP not_RB speak_VB for_IN the_DT big_JJ polluters_NNS ,_, but_CC who_WP speak_VBP for_IN all_DT of_
IN humanity_NN ,_, for_IN the_DT indigenous_JJ people_NNS of_IN the_DT world_NN ,_, for_IN the_DT bi
llions_NNS and_CC billions_NNS of_IN underprivileged_JJ people_NNS out_IN there_EX who_WP would_MD b
e_VB most_RBS affected_VBN by_IN this_DT ._. For_IN our_PRP$ children_NNS ’_VBP s_JJ children_NNS ,_
, and_CC for_IN those_DT people_NNS out_RP there_RB whose_WP$ voices_NNS have_VBP been_VBN drowned_V
BN out_RP by_IN the_DT politics_NNS of_IN greed_NN ._. I_PRP thank_VBP you_PRP all_DT for_IN this_DT
amazing_JJ award_NN tonight_NN ._. Let_VB us_PRP not_RB take_VB this_DT planet_NN for_IN granted_VBN
._. I_PRP do_VBP not_RB take_VB tonight_NN for_IN granted_VBN ._. Thank_NNP you_PRP so_RB very_RB mu
ch_JJ ._.
```

POS Tag Meanings : Here are the meanings of the Parts-Of-Speech tags used in NLTK

CC - Coordinating conjunction

CD - Cardinal number

DT - Determiner

EX - Existential there

FW - Foreign word

IN - Preposition or subordinating conjunction

JJ - Adjective

JJR - Adjective, comparative

JJS - Adjective, superlative

LS - List item marker

MD - Modal

NN - Noun, singular or mass

NNS - Noun, plural

NNP - Proper noun, singular

NNPS - Proper noun, plural

PDT - Predeterminer

POS - Possessive ending

PRP - Personal pronoun

PRP$ - Possessive pronoun

RB - Adverb

RBR - Adverb, comparative

RBS - Adverb, superlative

RP - Particle

SYM - Symbol

TO - to

UH - Interjection

VB - Verb, base form

VBD - Verb, past tense

VBG - Verb, gerund or present participle

VBN - Verb, past participle

VBP - Verb, non-3rd person singular present

VBZ - Verb, 3rd person singular present

WDT - Wh-determiner

WP - Wh-pronoun

WP$ -- Possessive wh-pronoun

WRB - Wh-adverb

Bag Of Words

In [19]:

```
dataset = nltk.sent_tokenize(paragraph)
for i in range(len(dataset)):
    dataset[i] = dataset[i].lower() # converted all the words in lower cae
    dataset[i] = re.sub(r'\W',' ',dataset[i]) #replace non-word by space
    dataset[i] = re.sub(r'\s+',' ',dataset[i]) # replace the all space by single space
```

In [20]:

```
print(dataset)
```

```
['thank you all so very much ', 'thank you to the academy ', 'thank you to all of you in this room '
, 'i have to congratulate the other incredible nominees this year ', 'the revenant was the product o
f the tireless efforts of an unbelievable cast and crew ', 'first off to my brother in this endeavor
mr tom hardy ', 'tom your talent on screen can only be surpassed by your friendship off screen thank
you for creating a transcendent cinematic experience ', 'thank you to everybody at fox and new regen
cy my entire team ', 'i have to thank everyone from the very onset of my career to my parents none o
f this would be possible without you ', 'and to my friends i love you dearly you know who you are ',
'and lastly i just want to say this making the revenant was about man s relationship to the natural
world ', 'a world that we collectively felt in 2015 as the hottest year in recorded history ', 'our
production needed to move to the southern tip of this planet just to be able to find snow climate ch
ange is real it is happening right now ', 'it is the most urgent threat facing our entire species an
d we need to work collectively together and stop procrastinating ', 'we need to support leaders arou
nd the world who do not speak for the big polluters but who speak for all of humanity for the indige
nous people of the world for the billions and billions of underprivileged people out there who would
be most affected by this ', 'for our children s children and for those people out there whose voices
have been drowned out by the politics of greed ', 'i thank you all for this amazing award tonight ',
'let us not take this planet for granted ', 'i do not take tonight for granted ', 'thank you so very
much ']
```

In [21]:

```
print(type(dataset))
```

```
<class 'list'>
```

In [22]:

```
#create histogram
word2count = {}
for data in dataset: # here data returns the list of sentence
    words = nltk.word_tokenize(data)#tokenize the list of sentences into words
    for word in words: # using for loop, we are taking each word from words
        if word not in word2count.keys(): # if word is not in word2count set then we are counting as first word
            word2count[word] = 1 # and setting as 1
        else:
            word2count[word] += 1 # if word is already contain in word2count then we are increasing the word occu
rence
```

In [23]:

```
print(word2count)
```

```
{'thank': 8, 'you': 12, 'all': 4, 'so': 2, 'very': 3, 'much': 2, 'to': 16, 'the': 17, 'academy': 1,
'of': 10, 'in': 4, 'this': 9, 'room': 1, 'i': 6, 'have': 3, 'congratulate': 1, 'other': 1, 'incredib
le': 1, 'nominees': 1, 'year': 2, 'revenant': 2, 'was': 2, 'product': 1, 'tireless': 1, 'efforts': 1
, 'an': 1, 'unbelievable': 1, 'cast': 1, 'and': 8, 'crew': 1, 'first': 1, 'off': 2, 'my': 5, 'brothe
r': 1, 'endeavor': 1, 'mr': 1, 'tom': 2, 'hardy': 1, 'your': 2, 'talent': 1, 'on': 1, 'screen': 2, '
can': 1, 'only': 1, 'be': 4, 'surpassed': 1, 'by': 3, 'friendship': 1, 'for': 10, 'creating': 1, 'a'
: 2, 'transcendent': 1, 'cinematic': 1, 'experience': 1, 'everybody': 1, 'at': 1, 'fox': 1, 'new': 1
, 'regency': 1, 'entire': 2, 'team': 1, 'everyone': 1, 'from': 1, 'onset': 1, 'career': 1, 'parents'
: 1, 'none': 1, 'would': 2, 'possible': 1, 'without': 1, 'friends': 1, 'love': 1, 'dearly': 1, 'know
': 1, 'who': 4, 'are': 1, 'lastly': 1, 'just': 2, 'want': 1, 'say': 1, 'making': 1, 'about': 1, 'man
': 1, 's': 2, 'relationship': 1, 'natural': 1, 'world': 4, 'that': 1, 'we': 3, 'collectively': 2, 'f
elt': 1, '2015': 1, 'as': 1, 'hottest': 1, 'recorded': 1, 'history': 1, 'our': 3, 'production': 1, '
needed': 1, 'move': 1, 'southern': 1, 'tip': 1, 'planet': 2, 'able': 1, 'find': 1, 'snow': 1, 'clima
te': 1, 'change': 1, 'is': 3, 'real': 1, 'it': 2, 'happening': 1, 'right': 1, 'now': 1, 'most': 2, '
urgent': 1, 'threat': 1, 'facing': 1, 'species': 1, 'need': 2, 'work': 1, 'together': 1, 'stop': 1,
'procrastinating': 1, 'support': 1, 'leaders': 1, 'around': 1, 'do': 2, 'not': 3, 'speak': 2, 'big':
1, 'polluters': 1, 'but': 1, 'humanity': 1, 'indigenous': 1, 'people': 3, 'billions': 2, 'underprivi
leged': 1, 'out': 3, 'there': 2, 'affected': 1, 'children': 2, 'those': 1, 'whose': 1, 'voices': 1,
'been': 1, 'drowned': 1, 'politics': 1, 'greed': 1, 'amazing': 1, 'award': 1, 'tonight': 2, 'let': 1
, 'us': 1, 'take': 2, 'granted': 2}
```

Find most frequent words

In [24]:

```
import heapq
```

In [25]:

```
freq_words = heapq.nlargest(100,word2count,key = word2count.get)
```

In [26]:

```
print(freq_words)
```

```
['the', 'to', 'you', 'of', 'for', 'this', 'thank', 'and', 'i', 'my', 'all', 'in', 'be', 'who', 'worl
d', 'very', 'have', 'by', 'we', 'our', 'is', 'not', 'people', 'out', 'so', 'much', 'year', 'revenant
', 'was', 'off', 'tom', 'your', 'screen', 'a', 'entire', 'would', 'just', 's', 'collectively', 'plan
et', 'it', 'most', 'need', 'do', 'speak', 'billions', 'there', 'children', 'tonight', 'take', 'grant
ed', 'academy', 'room', 'congratulate', 'other', 'incredible', 'nominees', 'product', 'tireless', 'e
fforts', 'an', 'unbelievable', 'cast', 'crew', 'first', 'brother', 'endeavor', 'mr', 'hardy', 'talen
t', 'on', 'can', 'only', 'surpassed', 'friendship', 'creating', 'transcendent', 'cinematic', 'experi
ence', 'everybody', 'at', 'fox', 'new', 'regency', 'team', 'everyone', 'from', 'onset', 'career', 'p
arents', 'none', 'possible', 'without', 'friends', 'love', 'dearly', 'know', 'are', 'lastly', 'want'
]
```

In [27]:

```
X = []
for data in dataset:
    vector = []
    for word in freq_words:
        #print(word)
        if word in nltk.word_tokenize(data):
            vector.append(1)
        else:
            vector.append(0)
    X.append(vector)
```

In [28]:

```
print(X)
```

```
[[0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
, [1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
], [0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1
, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
0], [0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0,
0], [1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0,
0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1], [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0], [0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0]]
```

We found the result as list of list. We need to convet it into 2D array

In [29]:

```
X = np.asarray(X)
```

In [30]:

```
# got the bag of words matrix
print(X)
```

```
[[0 0 1 ... 0 0 0]
 [1 1 1 ... 0 0 0]
 [0 1 1 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]]
```

In [31]:

```
print(X.shape)
```

```
(20, 100)
```

TF-IDF

TF stands for "Term Frequency"

Formula :

TF : (Number of occurrences of a word in the document)/(total number of word in that document)

Example:

"to be or not to be" if we calculate TF for word to,be,or then

1. for "to" = 2/6 = 0.33
2. for "be" = 2/6 = 0.33
3. for "or" = 1/6 = 0.16

IDF stands for Inverse Document Frequency

Formula:

IDF = loge(Number of documents)/(Number of documents containing word)

Example:

doc1 : "to be or not to be"

doc2 : i have to be

doc3 : you got to be

1. IDF for word "to" = log(e)(3/3) = 0
2. IDF for word "be" = log(e)(3/3) = 0
3. IDF for word "or" = log(e)(3/1) = 0.477

In [32]:

```
word_idfs = {} # take a sets word_idfs
for word in freq_words: # take each word from freq_words
    doc_count = 0
    for data in dataset: # take the each sentences from dataset
        words = nltk.word_tokenize(data)# coverted to word_tokenize
        if word in words:#check the frequent word from words if it ? we are counting 1
            doc_count += 1
    word_idfs[word] = np.log(len(dataset)/doc_count+1)#calculating the IDF values for each frequent words
```

In [33]:

```
print(word_idfs)
```

```
{'the': 1.0986122886681098, 'to': 1.0360919316867758, 'you': 1.1700712526502546, 'of': 1.4663370687934272, 'for': 1.4663370687934272, 'this': 1.1700712526502546, 'thank': 1.252762968495368, 'and': 1.349926716949016, 'i': 1.4663370687934272, 'my': 1.791759469228055, 'all': 1.791759469228055, 'in': 2.03688192726104, 'be': 1.791759469228055, 'who': 2.3978952727983707, 'world': 2.03688192726104, 'very': 2.03688192726104, 'have': 2.03688192726104, 'by': 2.03688192726104, 'we': 2.03688192726104, 'our': 2.03688192726104, 'is': 2.3978952727983707, 'not': 2.03688192726104, 'people': 2.3978952727983707, 'out': 2.3978952727983707, 'so': 2.3978952727983707, 'much': 2.3978952727983707, 'year': 2.3978952727983707, 'revenant': 2.3978952727983707, 'was': 2.3978952727983707, 'off': 2.3978952727983707, 'tom': 2.3978952727983707, 'your': 3.044522437723423, 'screen': 3.044522437723423, 'a': 2.3978952727983707, 'entire': 2.3978952727983707, 'would': 2.3978952727983707, 'just': 2.3978952727983707, 's': 2.3978952727983707, 'collectively': 2.3978952727983707, 'planet': 2.3978952727983707, 'it': 2.3978952727983707, 'most': 2.3978952727983707, 'need': 2.3978952727983707, 'do': 2.3978952727983707, 'speak': 3.044522437723423, 'billions': 3.044522437723423, 'there': 2.3978952727983707, 'children': 3.044522437723423, 'tonight': 2.3978952727983707, 'take': 2.3978952727983707, 'granted': 2.3978952727983707, 'academy': 3.044522437723423, 'room': 3.044522437723423, 'congratulate': 3.044522437723423, 'other': 3.044522437723423, 'incredible': 3.044522437723423, 'nominees': 3.044522437723423, 'product': 3.044522437723423, 'tireless': 3.044522437723423, 'efforts': 3.044522437723423, 'an': 3.044522437723423, 'unbelievable': 3.044522437723423, 'cast': 3.044522437723423, 'crew': 3.044522437723423, 'first': 3.044522437723423, 'brother': 3.044522437723423, 'endeavor': 3.044522437723423, 'mr': 3.044522437723423, 'hardy': 3.044522437723423, 'talent': 3.044522437723423, 'on': 3.044522437723423, 'can': 3.044522437723423, 'only': 3.044522437723423, 'surpassed': 3.044522437723423, 'friendship': 3.044522437723423, 'creating': 3.044522437723423, 'transcendent': 3.044522437723423, 'cinematic': 3.044522437723423, 'experience': 3.044522437723423, 'everybody': 3.044522437723423, 'at': 3.044522437723423, 'fox': 3.044522437723423, 'new': 3.044522437723423, 'regency': 3.044522437723423, 'team': 3.044522437723423, 'everyone': 3.044522437723423, 'from': 3.044522437723423, 'onset': 3.044522437723423, 'career': 3.044522437723423, 'parents': 3.044522437723423, 'none': 3.044522437723423, 'possible': 3.044522437723423, 'without': 3.044522437723423, 'friends': 3.044522437723423, 'love': 3.044522437723423, 'dearly': 3.044522437723423, 'know': 3.044522437723423, 'are': 3.044522437723423, 'lastly': 3.044522437723423, 'want': 3.044522437723423}
```

TF - Matrix

In [34]:

```
tf_matrix = {}
for word in freq_words:
    doc_tf = []
    for data in dataset:
        frequency = 0
        for w in nltk.word_tokenize(data):
            if word==w:
                frequency+=1
        tf_words = frequency/(len(nltk.word_tokenize(data)))
        doc_tf.append(tf_words)
    tf_matrix[word] = doc_tf
```

In [35]:

```
print(tf_matrix)
```

```
{'the': [0.0, 0.2, 0.0, 0.1, 0.2, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.1, 0.06666666666666667, 0.03571428571428571, 0.05, 0.10638297872340426, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'to': [0.0, 0.2, 0.1111111111111111, 0.1, 0.0, 0.09090909090909091, 0.0, 0.08333333333333333, 0.08695652173913043, 0.07692307692307693, 0.1, 0.0, 0.14285714285714285, 0.05, 0.02127659574468085, 0.0, 0.0, 0.0, 0.0, 0.0], 'you': [0.16666666666666666, 0.2, 0.2222222222222222, 0.0, 0.0, 0.0, 0.045454545454545456, 0.08333333333333333, 0.043478260869565216, 0.23076923076923078, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1111111111111111, 0.0, 0.0, 0.2], 'of': [0.0, 0.0, 0.1111111111111111, 0.0, 0.13333333333333333, 0.0, 0.0, 0.0, 0.08695652173913043, 0.0, 0.0, 0.0, 0.03571428571428571, 0.0, 0.06382978723404255, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'for': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0851063829787234, 0.09090909090909091, 0.1111111111111111, 0.125, 0.14285714285714285, 0.0], 'this': [0.0, 0.0, 0.1111111111111111, 0.1, 0.0, 0.09090909090909091, 0.0, 0.0, 0.043478260869565216, 0.0, 0.05, 0.0, 0.03571428571428571, 0.0, 0.02127659574468085, 0.0, 0.1111111111111111, 0.125, 0.0, 0.0], 'thank': [0.16666666666666666, 0.2, 0.1111111111111111, 0.0, 0.0, 0.0, 0.045454545454545456, 0.08333333333333333, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1111111111111111, 0.0, 0.0, 0.2], 'and': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.08333333333333333, 0.0, 0.07692307692307693, 0.05, 0.0, 0.0, 0.1, 0.02127659574468085, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'i': [0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.07692307692307693, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1111111111111111, 0.0, 0.14285714285714285, 0.0], 'my': [0.0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.08333333333333333, 0.08695652173913043, 0.07692307692307693, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'all': [0.16666666666666666, 0.0, 0.1111111111111111, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0212765957446808, 0.0, 0.1111111111111111, 0.0, 0.0, 0.0], 'in': [0.0, 0.0, 0.1111111111111111, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13333333333333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'be': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.03571428571428571, 0.0, 0.02127659574468085, 0.0, 0.0, 0.0, 0.0, 0.0], 'who': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07692307692307693, 0.0, 0.0, 0.0, 0.0, 0.06382978723404255, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'world': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05, 0.0666666666666666
```

7, 0.0, 0.0, 0.0425531914893617, 0.0, 0.0, 0.0, 0.0, 0.0], 'very': [0.16666666666666666, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2]
, 'have': [0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'by': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545
456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.02127659574468085, 0.045454545454545456, 0.0, 0.0, 0.0, 0.
0], 'we': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.05, 0.
02127659574468085, 0.0, 0.0, 0.0, 0.0, 0.0], 'our': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.03571428571428571, 0.05, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'is': [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07142857142857142, 0.05, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0], 'not': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0212765
9574468085, 0.0, 0.0, 0.125, 0.14285714285714285, 0.0], 'people': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0425531914893617, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0],
'out': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.02127659574468085, 0
.09090909090909091, 0.0, 0.0, 0.0, 0.0], 'so': [0.16666666666666666, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2], 'much': [0.16666666666666666, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2], 'year': [
0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0], 'revenant': [0.0, 0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'was': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0,
0.0, 0.0, 0.0, 0.0, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'off': [0.0, 0.0, 0.0, 0.0,
0.0, 0.09090909090909091, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0], 'tom': [0.0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.045454545454545456, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'your': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.090
90909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'screen': [0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0], 'a': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0666666666
6666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'entire': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08
333333333333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'would': [0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.02127659574468085, 0.0
, 0.0, 0.0, 0.0, 0.0], 'just': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05, 0.0, 0.03571
428571428571, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 's': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.05, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'collectively': [0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0], 'planet': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.03571428571428571,
0.0, 0.0, 0.0, 0.0, 0.125, 0.0, 0.0], 'it': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.03571428571428571, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'most': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05, 0.02127659574468085, 0.0, 0.0, 0.0, 0.0, 0.0], 'need': [0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05, 0.02127659574468085, 0.0, 0.0,
0.0, 0.0, 0.0], 'do': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.02127
659574468085, 0.0, 0.0, 0.0, 0.14285714285714285, 0.0], 'speak': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0425531914893617, 0.0, 0.0, 0.0, 0.0, 0.0], 'billions': [0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0425531914893617, 0.0, 0.0, 0.0, 0
.0, 0.0], 'there': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.02127659
574468085, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0], 'children': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0], 'tonight': [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1111111111111111, 0.0,
0.14285714285714285, 0.0], 'take': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.125, 0.14285714285714285, 0.0], 'granted': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.125, 0.14285714285714285, 0.0], 'academy': [0.0,
0.2, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'roo
m': [0.0, 0.0, 0.1111111111111111, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0], 'congratulate': [0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'other': [0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'incredible': [0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'nominees': [0.0, 0.0, 0.0, 0.1, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'product': [0.0, 0.0
, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0], 'tireless': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'efforts': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'an': [0.0, 0.0, 0.0, 0.0, 0.066
66666666666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'unbelie
vable': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'cast': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'crew': [0.0, 0.0, 0.0, 0.0, 0.06666666666666667,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'first': [0.0, 0.0, 0.0,
0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
, 'brother': [0.0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'endeavor': [0.0, 0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'mr': [0.0, 0.0, 0.0, 0.0, 0.0, 0.09090
909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'hardy': [0.0,
0.0, 0.0, 0.0, 0.0, 0.09090909090909091, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0], 'talent': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'on': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'can': [0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'only': [
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0], 'surpassed': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'friendship': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045
454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'creating': [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0], 'transcendent': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'cinematic': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.045454
545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'experience': [0.0,

0.0, 0.0, 0.0, 0.0, 0.0, 0.045454545454545456, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0], 'everybody': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08333333333333333, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'at': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.083333333333
33333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'fox': [0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.08333333333333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'new': [
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08333333333333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0], 'regency': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08333333333333333, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'team': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.083333333
33333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'everyone': [0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
, 'from': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0], 'onset': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'career': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'parents': [0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0], 'none': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'possible': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0434
78260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'without': [0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.043478260869565216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
, 'friends': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07692307692307693, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'love': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0769230769230
7693, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'dearly': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.07692307692307693, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'know': [0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07692307692307693, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0], 'are': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07692307692307693, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'lastly': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 'want': [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.05, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]}

TF-IDF Matrix

TF- IDF will be multiplication of TF and IDF values

In [36]:

```
tfidf_matrix = []
for word in tf_matrix.keys():
    tf_idf = []
    for value in tf_matrix[word]:
        score = value*word_idfs[word]
        tf_idf.append(score)
    tfidf_matrix.append(tf_idf)
```

In [37]:

```
print(tfidf_matrix)
```

[[0.0, 0.21972245773362198, 0.0, 0.10986122886681099, 0.21972245773362198, 0.0, 0.0, 0.0, 0.04776575
1681222164, 0.0, 0.10986122886681099, 0.07324081924454065, 0.039236153166718205, 0.05493061443340549
5, 0.11687364773064998, 0.04993692221218681, 0.0, 0.0, 0.0, 0.0], [0.0, 0.20721838633735518, 0.11512
132574297508, 0.10360919316867759, 0.0, 0.09419017560788871, 0.0, 0.08634099430723131, 0.09009495058
145876, 0.07969937936052122, 0.10360919316867759, 0.0, 0.14801313309811082, 0.051804596584338794, 0.
022044509184825017, 0.0, 0.0, 0.0, 0.0, 0.0], [0.1950118754417091, 0.23401425053005093, 0.2600158339
222788, 0.0, 0.0, 0.0, 0.05318505693864794, 0.09750593772085454, 0.05087266315870672, 0.270016442919
28956, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1300079169611394, 0.0, 0.0, 0.23401425053005093], [0.0, 0.0,
0.16292634097704745, 0.0, 0.19551160917245697, 0.0, 0.0, 0.0, 0.12750757119942846, 0.0, 0.0, 0.0, 0.
052369181028336668, 0.0, 0.09359598311447406, 0.06665168494515578, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.06665168494515578, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1247946441526321, 0.1333
0336989031155, 0.16292634097704745, 0.1832921335991784, 0.20947672411334672, 0.0], [0.0, 0.0, 0.1300
079169611394, 0.11700712526502546, 0.0, 0.10637011387729588, 0.0, 0.0, 0.050872663158706 72, 0.0, 0.0
5850356263251273, 0.0, 0.041788259023223376, 0.0, 0.0248951330351118, 0.0, 0.1300079169611394, 0.146
25890658128182, 0.0, 0.0], [0.20879382808256133, 0.2505525936990736, 0.1391958853883742, 0.0, 0.0, 0
.0, 0.05694377129524401, 0.10439691404128067, 0.05446795515197252, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.1391958853883742, 0.0, 0.0, 0.2505525936990736], [0.0, 0.0, 0.0, 0.0, 0.08999511446326773, 0.0,
0.0, 0.11249389307908465, 0.0, 0.10384051668838584, 0.0674963358474508, 0.0, 0.0, 0.1349926716949016
, 0.028721845041468422, 0.06136030531586436, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.1466337068793427
, 0.0, 0.0, 0.0, 0.0, 0.06375378559971423, 0.11279515913795594, 0.07331685343967136, 0.0, 0.0, 0.0,
0.0, 0.0, 0.16292634097704745, 0.0, 0.20947672411334672, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.162887224
47527773, 0.0, 0.1493132891023379, 0.15580517123722218, 0.13782765147908116, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0], [0.2986265782046758, 0.0, 0.19908438546978388, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.03812254189846925, 0.0, 0.19908438546978388, 0.0, 0.0, 0.0], [0.0, 0
.0, 0.22632021414011555, 0.0, 0.0, 0.1851710842964582, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2715842569681387,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08144361223763887, 0.0, 0.
07790258561861109, 0.0, 0.0, 0.0, 0.0, 0.06399140961528767, 0.0, 0.03812254189846925, 0.0, 0.0, 0.0, 0.0,
0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1844534825229516, 0.0, 0.0, 0.0, 0.0, 0.153057
14507223642, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.10184409
636305201, 0.13579212848406935, 0.0, 0.0, 0.08667582669195915, 0.0, 0.0, 0.0, 0.0, 0.0], [0.33948032
121017335, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08856008379395826, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.40737638545220806], [0.0, 0.0, 0.0, 0.20368819272610403, 0.0, 0.0, 0.0, 0.0, 0.08
856008379395826, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0925855421482291, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0925855421482291, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.04333791334597958, 0.092

5855421482291, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.135792
12848406935, 0.0, 0.10184409636305201, 0.04333791334597958, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.07274578311646572, 0.10184409636305201, 0.0, 0.0925
855421482291, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.17
127823377131218, 0.11989476363991854, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.04333791334597958, 0.0, 0.0, 0.25461024090763, 0.290983132
4658629, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.102038096714
82428, 0.10899523967265322, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.05101904835741214, 0.21799047934530644, 0.0, 0.0, 0.0, 0.0], [0.39964921213306
176, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.479
57905455967414], [0.39964921213306176, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.47957905455967414], [0.0, 0.0, 0.0, 0.23978952727983707, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.15985968485322471, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0,
0.0, 0.15985968485322471, 0.0, 0.0, 0.0, 0.0, 0.0, 0.11989476363991854, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.15985968485322471, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1198947636399
1854, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.21799047934530644, 0
.10899523967265322, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0
, 0.0, 0.0, 0.21799047934530644, 0.10899523967265322, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.10899523967
265322, 0.0, 0.0, 0.0, 0.0, 0.15985968485322471, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.19982460606653088, 0.0, 0.0, 0.0, 0.0, 0.0, 0.11989476363991854, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.10425631620862481, 0.0, 0.0, 0.0,
0.0, 0.0, 0.05101904835741214, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.11989476363991854, 0.0, 0.08563911688565609, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.11989476363991854, 0.0, 0.0, 0.0, 0.0, 0.1089952396726530
22, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.15985968485322471
, 0.0, 0.11989476363991854, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.08563911688565609, 0.0, 0.0, 0.0, 0.0, 0.2997369090997633, 0.0, 0.0], [0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.08563911688565609, 0.11989476363991854, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.119894
76363991854, 0.05101904835741214, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.11989476363991854, 0.05101904835741214, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05101904835741214, 0.0, 0.0, 0.0,
0.34255646754262437, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
1295541462861031, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.1295541462861031, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.05101904835741214, 0.10899523967265322, 0.0, 0.0, 0.0, 0.0], [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0,
0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.266432
80808870784, 0.0, 0.34255646754262437, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.29973690909979633, 0.34255646754262437, 0.0], [0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.29973690909979633, 0.3425564675426
2437, 0.0], [0.0, 0.6089044875446846, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.3382802708581581, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.3044522437723423, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.3044522437723423, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.30
44522437723423, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.
0, 0.0, 0.0, 0.3044522437723423, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.20296816251489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.20296816251489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.20296816251489486, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.20296816251
489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0,
0.0, 0.20296816251489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
, [0.0, 0.0, 0.0, 0.0, 0.20296816251489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.20296816251489486, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.2767747670657657
6, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0,
0.27677476706576576, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.
0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.27677476706576576, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.138
38738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
[0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1383873835328
8288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.13838738353288288, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2537102031436186, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2537102031436186, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2537102031436186, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.253710
2031436186, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.2537102031436186, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0,

```
0.0, 0.0, 0.0, 0.0, 0.0, 0.2537102031436186, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1
323705407705836, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1323705407705836, 0.0, 0.0, 0.0, 0
.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.234194033671
03256, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0
.0, 0.23419403367103256, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0
, 0.0, 0.0, 0.0, 0.0, 0.23419403367103256, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.23419403367103256, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.23419403367103256, 0.0, 0.0, 0.0, 0.0, 0.
0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.15222612188617116,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1
5222612188617116, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]]
```

In [38]:

```
tfidf_matrix = np.asarray(tfidf_matrix)
```

In [39]:

```
print(tfidf_matrix)
```

```
[[0.         0.21972246 0.         ... 0.         0.         0.        ]
 [0.         0.20721839 0.11512133 ... 0.         0.         0.        ]
 [0.19501188 0.23401425 0.26001583 ... 0.         0.         0.23401425]
 ...
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]]
```

In [40]:

```
print(tfidf_matrix.shape)
```

```
(100, 20)
```

In [41]:

```
tfidf_matrix = np.transpose(tfidf_matrix)
```

In [42]:

```
print(tfidf_matrix)
```

```
[[0.         0.         0.19501188 ... 0.         0.         0.        ]
 [0.21972246 0.20721839 0.23401425 ... 0.         0.         0.        ]
 [0.         0.11512133 0.26001583 ... 0.         0.         0.        ]
 ...
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 [0.         0.         0.23401425 ... 0.         0.         0.        ]]
```

N-Gram Modeling

In [51]:

```
text = """he term is frequently used interchangeably with the term climate change, though the latter refers to
          both human- and naturally produced warming and the effects it has on our planet.
          It is most commonly measured as the average increase in Earth’s global surface temperature."""
```

In [44]:

```
n= 2
ngrams = {}
for i in range(len(text)-n):
    gram = text[i:i+n] # text[0:3]
    if gram not in ngrams.keys():
        ngrams[gram] = []
    ngrams[gram].append(text[i+n])
```

In [45]:

```
print(ngrams)
```

```
{'he': [' ', ' ', ' ', ' ', ' '], 'e ': ['t', 't', 'c', 'l', 'e', 'a', 'i', 'i', 't'], ' t': ['e', '
h', 'e', 'h', 'h', 'o', 'h', 'h', 'e'], 'te': ['r', 'r', 'r', ' ', 'r', 'm'], 'er': ['m', 'c', 'm',
' ', 's', 'a', 'a'], 'rm': [' ', ' ', 'i'], 'm ': ['i', 'c'], ' i': ['s', 'n', 't', 's', 'n', 'n'],
'is': [' ', ' '], 's ': ['f', 't', 'i', 'o', 'm', 't', 'g'], ' f': ['r'], 'fr': ['e'], 're': ['q', '
f', 'd', 'a', '.'], 'eq': ['u'], 'qu': ['e'], 'ue': ['n'], 'en': ['t'], 'nt': ['l', 'e'], 'tl': ['y'
], 'ly': [' ', ' ', ' ', ' '], 'y ': ['u', 'w', 'p', 'm'], ' u': ['s'], 'us': ['e'], 'se': ['d', ' '
], 'ed': [' ', ' ', ' '], 'd ': ['i', 'n', 'w', 't', 'a'], 'in': ['t', 'g', 'c', ' '], 'rc': ['h'],
'ch': ['a', 'a'], 'ha': ['n', 'n', 's'], 'an': ['g', 'g', '-', 'd', 'd', 'e'], 'ng': ['e', 'e', ' ']
, 'ge': ['a', ',', ' '], 'ea': ['b', 's', 's'], 'ab': ['l'], 'bl': ['y'], ' w': ['i', 'a'], 'wi': ['
t'], 'it': ['h', ' '], 'th': [' ', 'e', 'o', 'e', ' ', 'e', 'e', '’'], 'h ': ['t', 't', 'h'], ' c':
['l', 'h', 'o'], 'cl': ['i'], 'li': ['m'], 'im': ['a'], 'ma': ['t', 'n'], 'at': ['e', 't', 'u', 'u']
, 'e,': [' '], ', ': ['t'], 'ho': ['u'], 'ou': ['g', 'r'], 'ug': ['h'], 'gh': [' '], ' l': ['a'], 'l
a': ['t', 'n'], 'tt': ['e'], 'r ': ['r', 'p'], ' r': ['e'], 'ef': ['e', 'f'], 'fe': ['r', 'c'], 'rs'
: [' '], 'to': [' '], 'o ': ['\n'], ' \n': [' ', ' '], '\n ': [' ', ' '], '  ': [' ', ' ', ' ', ' ',
' ', ' ', ' ', ' ', ' ', 'b', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', 'I'], ' b': ['o'], 'bo': ['t'], 'ot
': ['h'], ' h': ['u', 'a'], 'hu': ['m'], 'um': ['a'], 'n-': [' '], '- ': ['a'], ' a': ['n', 'n', 's'
, 'v'], 'nd': [' ', ' '], ' n': ['a'], 'na': ['t'], 'tu': ['r', 'r'], 'ur': ['a', ' ', 'e', 'f', 'e'
], 'ra': ['l', 'g', 't'], 'al': ['l', ' '], 'll': ['y'], ' p': ['r', 'l'], 'pr': ['o'], 'ro': ['d'],
'od': ['u'], 'du': ['c'], 'uc': ['e'], 'ce': ['d', ' '], 'wa': ['r'], 'ar': ['m', 't'], 'mi': ['n'],
'g ': ['a'], ' e': ['f'], 'ff': ['e'], 'ec': ['t'], 'ct': ['s'], 'ts': [' '], 't ': ['h', 'i', 'c'],
'as': [' ', 'u', ' ', 'e'], ' o': ['n', 'u'], 'on': [' ', 'l'], 'n ': ['o', 'E'], 'pl': ['a'], 'ne':
['t'], 'et': ['.'], 't.': [' '], '. ': ['\n'], ' I': ['t'], 'It': [' '], ' m': ['o', 'e'], 'mo': ['s
', 'n'], 'os': ['t'], 'st': [' '], 'co': ['m'], 'om': ['m'], 'mm': ['o'], 'nl': ['y'], 'me': ['a'],
'su': ['r', 'r'], 'av': ['e'], 've': ['r'], 'ag': ['e'], 'nc': ['r'], 'cr': ['e'], ' E': ['a'], 'Ea'
: ['r'], 'rt': ['h'], 'h’': ['s'], '’s': [' '], ' g': ['l'], 'gl': ['o'], 'lo': ['b'], 'ob': ['a'],
'ba': ['l'], 'l ': ['s'], ' s': ['u'], 'rf': ['a'], 'fa': ['c'], 'ac': ['e'], 'em': ['p'], 'mp': ['e
'], 'pe': ['r']}
```

In [52]:

```
ngrams1 = {}
words = nltk.word_tokenize(text)
for i in range(len(words)-n):
    gram = ' '.join(words[i:i+n])
    if gram not in ngrams1.keys():
        ngrams1[gram] = []
    ngrams1[gram].append(words[i+n])
```

In [53]:

```
print(ngrams1)
```

```
{'he term': ['is'], 'term is': ['frequently'], 'is frequently': ['used'], 'frequently used': ['inter
changeably'], 'used interchangeably': ['with'], 'interchangeably with': ['the'], 'with the': ['term'
], 'the term': ['climate'], 'term climate': ['change'], 'climate change': [','], 'change ,': ['thoug
h'], ', though': ['the'], 'though the': ['latter'], 'the latter': ['refers'], 'latter refers': ['to'
], 'refers to': ['both'], 'to both': ['human-'], 'both human-': ['and'], 'human- and': ['naturally']
, 'and naturally': ['produced'], 'naturally produced': ['warming'], 'produced warming': ['and'], 'wa
rming and': ['the'], 'and the': ['effects'], 'the effects': ['it'], 'effects it': ['has'], 'it has':
['on'], 'has on': ['our'], 'on our': ['planet'], 'our planet': ['.'], 'planet .': ['It'], '. It': ['
is'], 'It is': ['most'], 'is most': ['commonly'], 'most commonly': ['measured'], 'commonly measured'
: ['as'], 'measured as': ['the'], 'as the': ['average'], 'the average': ['increase'], 'average incre
ase': ['in'], 'increase in': ['Earth'], 'in Earth': ['’'], 'Earth ’': ['s'], '’ s': ['global'], 's g
lobal': ['surface'], 'global surface': ['temperature'], 'surface temperature': ['.']}
```

Latent Semantic Analysis

In [58]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import TruncatedSVD

#sample data
dataset = ["The amount of polution is increasing day by daya",
           "The concert was just great",
           "I Love to see Garden Ramsay Cook",
            "Google is introducting a new technology",
            "AI Robots are examples of great technology present to you",
            "All of us were singing in the concert",
            "We have Launch compaigns to stop pollution and global warming"]

dataset = [line.lower() for line in dataset]

vectorizer = TfidfVectorizer()
x = vectorizer.fit_transform(dataset)
print(x[0])
```

```
  (0, 10)       0.3604707823245737
  (0, 5)        0.3604707823245737
  (0, 9)        0.3604707823245737
  (0, 18)       0.3604707823245737
  (0, 20)       0.29922170630677863
  (0, 27)       0.3604707823245737
  (0, 25)       0.25576479528730944
  (0, 2)        0.3604707823245737
  (0, 35)       0.25576479528730944
```

Observation : The tfidf value of The is 0.3604707823245737. The 0 indicates the first row and 10 indicates the position of The. Similary for all the words for first document

In [60]:

```
print(x[1])
```

```
  (0, 15)       0.4213298560187446
  (0, 21)       0.5075738143811802
  (0, 39)       0.5075738143811802
  (0, 7)        0.4213298560187446
  (0, 35)       0.36013879374975194
```

Observation : The tfidf value of The is 0.3604707823245737. The 0 indicates the first row and 10 indicates the position of The. Similary for all the words for first document

In [66]:

```
lsa = TruncatedSVD(n_components = 4,n_iter=100) # we have taken the 4 components
lsa.fit(x)


terms = vectorizer.get_feature_names()
for i,comp in enumerate(lsa.components_):
    componentTerms = zip(terms,comp)
    sortedTerms = sorted(componentTerms,key=lambda x:x[1],reverse=True)
    sortedTerms = sortedTerms[:10]
    print("\nconcept",i,":")
    for term in sortedTerms:
        print(term)
```

```
concept 0 :
('the', 0.37931274666161524)
('concert', 0.3322524212584805)
('of', 0.30297933562408524)
('great', 0.2883406330890268)
('just', 0.22747274585828073)
('was', 0.22747274585828073)
('is', 0.1938158804638517)
('technology', 0.18182537167789728)
('all', 0.17278996134480654)
('in', 0.17278996134480654)

concept 1 :
('to', 0.3549401919583129)
('technology', 0.23931386858616868)
('cook', 0.21500250606772944)
('garden', 0.21500250606772944)
('love', 0.21500250606772944)
('ramsay', 0.21500250606772944)
('see', 0.21500250606772944)
('google', 0.15506130357728057)
('introducting', 0.15506130357728057)
('new', 0.15506130357728057)

concept 2 :
('is', 0.3648279017955781)
('google', 0.3151297909758329)
('introducting', 0.3151297909758329)
('new', 0.3151297909758329)
('technology', 0.26764814808496595)
('amount', 0.12437642265177631)
('by', 0.12437642265177631)
('day', 0.12437642265177631)
('daya', 0.12437642265177631)
('increasing', 0.12437642265177631)

concept 3 :
('and', 0.2549178847054724)
('compaigns', 0.25491788470547233)
('global', 0.25491788470547233)
('have', 0.25491788470547233)
('launch', 0.25491788470547233)
('pollution', 0.25491788470547233)
('stop', 0.25491788470547233)
('warming', 0.25491788470547233)
('we', 0.25491788470547233)
('by', 0.11840170938221117)
```

Word Synonyms and Antonyms using NLTK

In [69]:

```
from nltk.corpus import wordnet

synonyms = []
antonyms = []

for syn in wordnet.synsets("good"):
    for s in syn.lemmas():
        synonyms.append(s.name())
        for a in s.antonyms():
            antonyms.append(a.name())
```

In [71]:

```
print(set(synonyms))
print(set(antonyms))
```

```
{'commodity', 'unspoiled', 'just', 'good', 'goodness', 'dependable', 'full', 'undecomposed', 'skilfu
l', 'expert', 'right', 'in_force', 'trade_good', 'upright', 'skillful', 'estimable', 'honorable', 'w
ell', 'practiced', 'safe', 'in_effect', 'near', 'unspoilt', 'dear', 'honest', 'soundly', 'proficient
', 'beneficial', 'ripe', 'adept', 'salutary', 'effective', 'serious', 'respectable', 'sound', 'thoro
ughly', 'secure'}
{'ill', 'badness', 'bad', 'evil', 'evilness'}
```

In [ ]: