:[2] In

```
#import the necessary libraries
import pandas as pd
import numpy as np
import nltk
nltk.download('stopwords')
nltk.download('punkt')
```

```
nltk_data] Downloading package stopwords to]
...nltk_data]     C:\Users\D7me_\AppData\Roaming\nltk_data]
!nltk_data]   Package stopwords is already up-to-date]
nltk_data] Downloading package punkt to]
...nltk_data]     C:\Users\D7me_\AppData\Roaming\nltk_data]
!nltk_data]   Package punkt is already up-to-date]
```

Out[2]:

```
True
```

:[3] In

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
import os
import string
import copy
import pickle
```

:[5] In

```
title = "comp.graphics"
os.chdir(".\\mini_newsgroups")
paths = []
for (dirpath, dirnames, filenames) in os.walk(str(os.getcwd())+'/'+title+'/'):
    for i in filenames:
        paths.append(str(dirpath)+str("\\")+i)
print(dirpath)
```

```
/C:\Users\D7me_\mini_newsgroups\mini_newsgroups/comp.graphics
```

:[6] In

```
paths[0]
```

Out[6]:

```
'C:\\Users\\D7me_\\mini_newsgroups\\mini_newsgroups/comp.graphics/\\37916'
```

In [7]:

```
len(paths)
```

Out[7]:

```
100
```

In [8]:

```
postings = pd.DataFrame()
frequency = pd.DataFrame()
```

:[9] In

```
def remove_stop_words(data):
    stop_words = stopwords.words('english')
    words = word_tokenize(str(data))
    new_text = ""
    for w in words:
        if w not in stop_words:
            new_text = new_text + " " + w
    return np.char.strip(new_text)

#Removing punctuation
def remove_punctuation(data):
    symbols = "!\"#$%&()*+-./:;<=>?@[\]^_`{|}~\n"
    for i in range(len(symbols)):
        data = np.char.replace(data, symbols[i], ' ')
        data = np.char.replace(data, "  ", " ")
    data = np.char.replace(data, ',', '')
    return data

#Convert to lowercase
def convert_lower_case(data):
    return np.char.lower(data)

#Stemming
def stemming(data):
    stemmer= PorterStemmer()

    tokens = word_tokenize(str(data))
    new_text = ""
    for w in tokens:
        new_text = new_text + " " + stemmer.stem(w)
    return np.char.strip(new_text)
def convert_numbers(data):
    data = np.char.replace(data, "0", " zero ")
    data = np.char.replace(data, "1", " one ")
    data = np.char.replace(data, "2", " two ")
    data = np.char.replace(data, "3", " three ")
    data = np.char.replace(data, "4", " four ")
    data = np.char.replace(data, "5", " five ")
    data = np.char.replace(data, "6", " six ")
    data = np.char.replace(data, "7", " seven ")
    data = np.char.replace(data, "8", " eight ")
    data = np.char.replace(data, "9", " nine ")
    return data

#Removing header
def remove_header(data):
    try:
        ind = data.index('\n\n')
        data = data[ind:]
    except:
        print("No Header")
    return data

#Removing apostrophe
def remove_apostrophe(data):
    return np.char.replace(data, "'", "")

#Removing single characters
def remove_single_characters(data):
```

```
    words = word_tokenize(str(data))
    new_text = ""
    for w in words:
        if len(w) > 1:
            new_text = new_text + " " + w
    return np.char.strip(new_text)
```

:[10] In

```
def preprocess(data, query):
    if not query:
        data = remove_header(data)
        data = convert_lower_case(data)
        data = convert_numbers(data)
        data = remove_punctuation(data)
        data = remove_stop_words(data)
        data = remove_apostrophe(data)
        data = remove_single_characters(data)
        data = stemming(data)
    return data
```

:[11] In

```
doc = 0
for path in paths:
    file = open(path, 'r', encoding='cp1250')
    text = file.read().strip()
    file.close()
    preprocessed_text = preprocess(text, False)
    if doc%100 == 0:
        print(doc)
    tokens = word_tokenize(str(preprocessed_text))

    pos = 0
    for token in tokens:
        if token in postings:
            p = postings[token][0]

            k = [a[0] for a in p]
            if doc in k:
                for a in p:
                    if a[0] == doc:
                        a[1].add(pos)
            else:
                p.append([doc,{pos}])
                frequency[token][0] += 1
        else:
            postings.insert(value=[[[doc, {pos}]]], loc=0, column=token)
            frequency.insert(value=[1], loc=0, column=token)

        pos += 1
    doc += 1
```

```
0
```

In [12]:

```
postings
```

Out[12]:

l	sipp	simpl	...	truetyp	pete	ventur	undercut	nasti	shop	smirk	rip	lazer	fontmong	
]		,0]]												
}		,[{9}												
]		,1]												
}	,0]]	,[{47}		,99]]										
]	,8}	,15]	...	,76 ,57}	,99]]	,99]]	,99]]	,99]]	,99]]	,99]]	,99]]	,99]]	[[{98} ,99]]	0
}	[[{29	,[{8}		[[{31	[[{38}	[[{60}	[[{61}	[[{64}	[[{71}	[[{72}	[[{81}	[[{92}		
]		,17]												
}		,1080}												
.		...4												

rows × 4821 columns 1

In [13]:

```
postings["call"]
```

Out[13]:

```
... ,5378} ,17] ,[{78} ,10] ,[{43} ,5] ,[{7} ,0]]    0
Name: call, dtype: object
```

In [14]:

```
postings.to_pickle(title + "_positional_postings")
```

In [15]:

```
frequency.to_pickle(title + "_positional_frequency")
```

In [16]:

```
postings = pd.read_pickle(title + "_positional_postings")
```

In [17]:

```
frequency = pd.read_pickle(title + "_positional_frequency")
```

:[18] In

```
def get_word_postings(word):
    preprocessed_word = str(preprocess(word, True))
    print(preprocessed_word)
    print("Frequency:",frequency[preprocessed_word][0])
    print("Postings List:",postings[preprocessed_word][0])

def get_positions(posting_values, doc):
    for posting_value in posting_values:
        if posting_value[0] == doc:
            return posting_value[1]
    return {}

def gen_init_set_matchings(word):
    init = []
    word_postings = postings[word][0]
    for word_posting in word_postings:
        for positions in word_posting[1]:
            init.append((word_posting[0], positions))
    return init

def match_positional_index(init, b):
    matched_docs = []
    for p in init:
        doc = p[0]
        pos = p[1]

        count = 0

        for k in b:
            pos = pos+1
            k_pos = postings[k][0]
            docs_list = [z[0] for z in k_pos]
            if doc in docs_list:
                doc_positions = get_positions(k_pos, doc)
                if pos in doc_positions:
                    count += 1
                else:
                    count += 1
                    break

            if count == len(b):
                matched_docs.append(p[0])
    return set(matched_docs)

def run_query(query):
    processed_query = preprocess(query, True)
    print(processed_query)

    query_tokens = word_tokenize(str(processed_query))
    print(query_tokens)

    if len(query_tokens)==1:
        print("Total Document Mathces", [a[0] for a in postings[query][0]])
        return [a[0] for a in postings[query][0]]

    init_word = query_tokens[0]
    init_matches = gen_init_set_matchings(init_word)
```

```
    query_tokens.pop(0)
    total_matched_docs = match_positional_index(init_matches, query_tokens)
    print("Total Document Matches:", total_matched_docs)
    return total_matched_docs
```

:[19] In

```
query = "routin"

lists = run_query(query)
```

```
routin
['routin']
[Total Document Mathces [0, 5, 10, 16, 17, 49, 64, 91, 96
```

:[20] In

```
get_word_postings("call")
```

```
call
Frequency: 9
 Postings List: [[0, {7}], [5, {43}], [10, {78}], [17, {5378, 3204, 393, 272,
 115, 4566, 3159}], [18, {38}], [64, {896, 5640, 7055, 4626, 4631, 151, 1433,
 4635, 3755, 1970, 6203, 187, 5202, 4436, 6620, 4446, 864, 6884, 5868, 6775,
5624, 764}], [66, {92}], [74, {34, 4}], [91, {3398, 4431, 6290, 3381, 403
[[{1
```

:[21] In

```
query = postings
lists = run_query(query)
```

```
\  fontmong         lazer           rip         smirk          shop
   [[{71} ,99]]  [[{72} ,99]]  [[{81} ,99]]  [[{92} ,99]]  [[{98} ,99]]  0

\  nasti      undercut        ventur          pete
   [[{38} ,99]]  [[{60} ,99]]  [[{61} ,99]]  [[{64} ,99]]  0

\  ...  truetyp
   ...  [[{31 ,76 ,57} ,99]]  0

\  simpl            sipp
   [[{29 ,8} ,0]]  ...4 ,1080} ,17] ,[{8} ,15] ,[{47} ,1] ,[{9} ,0]]  0

\  call
   ... ,5378} ,17] ,[{78} ,10] ,[{43} ,5] ,[{7} ,0]]  0

\  routin
   ...1] ,[{17} ,10] ,[{129 ,120} ,5] ,[{6 ,33} ,0]]  0

\  render
   ...2} ,46] ,[{848} ,17] ,[{26} ,1] ,[{5 ,32} ,0]]  0

\  librari
   ...122} ,17] ,[{14} ,5] ,[{16} ,1] ,[{31 ,4} ,0]]  0

\  describ
   ...4} ,65] ,[{2349} ,64] ,[{1325} ,17] ,[{3} ,0]]  0

\  file
   ...,8] ,[{30 ,17} ,7] ,[{177 ,153} ,5] ,[{2} ,0]]  0

\  got
   ...[{112} ,37] ,[{57} ,33] ,[{54} ,29] ,[{1} ,0]]  0

  recent
  ... ,[{19} ,41] ,[{0} ,13] ,[{176} ,5] ,[{0} ,0]]  0

[rows x 4821 columns 1]
 fontmong', 'lazer', 'rip', 'smirk', 'shop', '\\', '0', '[', '[', '99',']
 ',', '{', '98', '}', ']', ']', '[', '[', '99', ',', '{', '92', '}', ']',
 ']', '[', '[', '99', ',', '{', '81', '}', ']', ']', '[', '[', '99', ',',
'{', '72', '}', ']', ']', '[', '[', '99', ',', '{', '71', '}', ']', ']', 'na
sti', 'undercut', 'ventur', 'pete', '\\', '0', '[', '[', '99', ',', '{', '6
 4', '}', ']', ']', '[', '[', '99', ',', '{', '61', '}', ']', ']', '[', '[',
 '99', ',', '{', '60', '}', ']', ']', '[', '[', '99', ',', '{', '38', '}',
 ']', ']', 'truetyp', '...', '\\', '0', '[', '[', '99', ',', '{', '57', ',',
 '76', ',', '31', '}', ']', ']', '...', 'simpl', 'sipp', '\\', '0', '[', '[',
 '0', ',', '{', '9', '}', ']', ',', '[', '1', ',', '{', '47', '}', ']', ',',
 '[', '15', ',', '{', '8', '}', ']', ',', '[', '17', ',', '{', '1080', ',',
 '4', '...', '[', '[', '0', ',', '{', '8', ',', '29', '}', ']', ']', 'call',
 '\\', '0', '[', '[', '0', ',', '{', '7', '}', ']', ',', '[', '5', ',', '{',
 '43', '}', ']', ',', '[', '10', ',', '{', '78', '}', ']', ',', '[', '17',
 ',', '{', '5378', ',', '...', 'routin', '\\', '0', '[', '[', '0', ',', '{',
 '33', ',', '6', '}', ']', ',', '[', '5', ',', '{', '120', ',', '129', '}',
']', ',', '[', '10', ',', '{', '17', '}', ']', ',', '[', '1', '...', 'rende
 r', '\\', '0', '[', '[', '0', ',', '{', '32', ',', '5', '}', ']', ',', '[',
```

```
 '1', ',', '{', '26', '}', ']', ',', '[', '17', ',', '{', '848', '}', ']',
 ',', '[', '46', ',', '{', '2', '...', 'librari', '\\', '0', '[', '[', '0',
 ',', '{', '4', ',', '31', '}', ']', ',', '[', '1', ',', '{', '16', '}', ']',
 ',', '[', '5', ',', '{', '14', '}', ']', ',', '[', '17', ',', '{', '122',
 '...', 'describ', '\\', '0', '[', '[', '0', ',', '{', '3', '}', ']', ',',
 '[', '17', ',', '{', '1325', '}', ']', ',', '[', '64', ',', '{', '2349',
 '}', ']', ',', '[', '65', ',', '{', '4', '...', 'file', '\\', '0', '[', '[',
 '0', ',', '{', '2', '}', ']', ',', '[', '5', ',', '{', '153', ',', '177',
 '}', ']', ',', '[', '7', ',', '{', '17', ',', '30', '}', ']', ',', '[', '8',
 ',', '...', 'got', '\\', '0', '[', '[', '0', ',', '{', '1', '}', ']', ',',
 '[', '29', ',', '{', '54', '}', ']', ',', '[', '33', ',', '{', '57', '}',
 ']', ',', '[', '37', ',', '{', '112', '}', ']', '...', 'recent', '0', '[',
 '[', '0', ',', '{', '0', '}', ']', ',', '[', '5', ',', '{', '176', '}', ']',
 ',', '[', '13', ',', '{', '0', '}', ']', ',', '[', '41', ',', '{', '19',
['[' ,'']', ']', ',', '...', '[', '1', 'rows', 'x', '4821', 'columns
()Total Document Matches: set
```