import re  
from collections import Counter, defaultdict  
import pandas as pd  
from Porter\_Stemmer\_Python import PorterStemmer  
  
  
def load\_file(filepath):  
 with open(filepath, 'r', encoding='utf-8') as file:  
 return file.read()  
  
#;oad paragraphs and stop words from given files  
textParagraphs = load\_file('Project4\_paragraphs.txt')  
stopWordstxt = set(load\_file('Project4\_stop\_words.txt').split())  
  
#initialize the stemmer  
stemmer = PorterStemmer()  
  
#tokenizing by paragraphs  
paragraphs = textParagraphs.split('\n\n') #assuming paragraphs are separated by double newlines i think  
  
  
def cleanText(text):  
 text = re.sub(r'<.\*?>', '', text) #remove HTML tags  
 text = re.sub(r'[^\w\s]', '', text) #remove punctuation  
 text = re.sub(r'\d+', '', text) #remove numbers  
 return text.lower() #convert to lowercase  
  
def removeStopWords(tokens):  
 return [word for word in tokens if word not in stopWordstxt]  
  
def wordStemmer(tokens):  
 return [stemmer.stem(word, 0, len(word) - 1) for word in tokens]  
  
#extract frequency and create feature vector  
paragraphWordCounts = []  
for paragraph in paragraphs:  
 cleanedParagraph = cleanText(paragraph)  
 tokens = cleanedParagraph.split() #further tokenize each paragraph into tokens  
 tokens = removeStopWords(tokens)   
 stemmedWords = wordStemmer(tokens)   
   
 #get word frequency for each paragraph  
 wordFreq = Counter(stemmedWords)  
 paragraphWordCounts.append(wordFreq)  
  
#aggregate word counts across all paragraphs  
totalWordFreq = Counter()  
for wordCount in paragraphWordCounts:  
 totalWordFreq.update(wordCount)  
  
T = 30 #threshold for feature vec  
  
#generate feature vector by selecting words that appear at least T times across all paragraphs  
featureVec = [word for word, count in totalWordFreq.items() if count >= T]  
  
print(f"Feature Vector (words with frequency >= {T}):", featureVec)  
  
tdm = []  
for wordCount in paragraphWordCounts:  
 row = [wordCount.get(word, 0) for word in featureVec]  
 tdm.append(row)  
  
tdm\_df = pd.DataFrame(tdm, columns=featureVec)  
print("\nTerm Document Matrix (TDM):")  
tdm\_df

Feature Vector (words with frequency >= 25): ['on', 'watch', 'thing', 'scene', 'go', 'us', 'more', 'out', 'well', 'film', 'veri', 'see', 'perform', 'great', 'life', 'realli', 'come', 'plai', 'wai', 'time', 'plot', 'charact', 'even', 'movi', 'look', 'end', 'up', 'make', 'much', 'work', 'seem', 'good', 'think', 'stori', 'take']  
  
Term Document Matrix (TDM):

|  | on | watch | thing | scene | go | us | more | out | well | film | ... | end | up | make | much | work | seem | good | think | stori | take |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 3 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 3 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | ... | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 4 | 6 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | ... | 0 | 0 | 1 | 0 | 1 | 2 | 2 | 0 | 0 | 0 |
| 5 | 2 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 1 | ... | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 4 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 2 | 0 | ... | 1 | 0 | 0 | 2 | 1 | 0 | 1 | 1 | 0 | 0 |
| 7 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 6 | 2 | 0 | ... | 3 | 5 | 0 | 1 | 2 | 0 | 2 | 1 | 1 | 1 |
| 9 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | ... | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 0 | ... | 0 | 2 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 2 | 4 | 1 | 1 | 0 | 0 | 2 | ... | 0 | 0 | 3 | 0 | 2 | 0 | 1 | 0 | 0 | 1 |
| 12 | 4 | 0 | 0 | 3 | 0 | 0 | 1 | 0 | 0 | 1 | ... | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 13 | 3 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | ... | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 14 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 6 | ... | 2 | 0 | 1 | 1 | 0 | 2 | 2 | 0 | 3 | 0 |
| 15 | 0 | 0 | 1 | 0 | 1 | 4 | 0 | 1 | 1 | 0 | ... | 0 | 2 | 0 | 2 | 0 | 0 | 1 | 0 | 3 | 2 |
| 16 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | ... | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 17 | 2 | 0 | 2 | 0 | 0 | 3 | 2 | 0 | 0 | 12 | ... | 1 | 0 | 7 | 1 | 0 | 4 | 1 | 0 | 0 | 0 |
| 18 | 1 | 0 | 0 | 0 | 2 | 1 | 1 | 3 | 1 | 6 | ... | 0 | 1 | 0 | 1 | 4 | 1 | 3 | 2 | 0 | 0 |
| 19 | 0 | 2 | 0 | 1 | 3 | 0 | 1 | 2 | 0 | 4 | ... | 0 | 1 | 3 | 1 | 2 | 1 | 0 | 0 | 3 | 0 |
| 20 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 7 | ... | 1 | 0 | 3 | 0 | 0 | 3 | 0 | 4 | 1 | 1 |
| 21 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 3 | ... | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 22 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | ... | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 23 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | ... | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 24 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 9 | ... | 1 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 |
| 25 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 3 | 2 |
| 26 | 1 | 0 | 2 | 0 | 0 | 0 | 1 | 1 | 0 | 2 | ... | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 27 | 0 | 0 | 0 | 0 | 2 | 1 | 2 | 0 | 0 | 7 | ... | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 28 | 1 | 2 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | ... | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 29 | 2 | 0 | 0 | 3 | 0 | 1 | 0 | 1 | 0 | 4 | ... | 0 | 1 | 2 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 30 | 2 | 4 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | ... | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 31 | 1 | 0 | 1 | 0 | 3 | 0 | 1 | 0 | 0 | 0 | ... | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 2 | 0 | 0 |
| 32 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | ... | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 2 |
| 33 | 2 | 1 | 2 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | ... | 0 | 0 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 1 |
| 34 | 4 | 2 | 0 | 3 | 1 | 1 | 3 | 4 | 2 | 9 | ... | 2 | 3 | 3 | 1 | 3 | 2 | 2 | 1 | 4 | 2 |
| 35 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 5 | ... | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 36 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 1 | 3 | 0 | 0 | 0 | 0 | 2 | ... | 0 | 0 | 3 | 2 | 0 | 0 | 2 | 1 | 1 | 2 |
| 38 | 3 | 0 | 2 | 0 | 0 | 0 | 2 | 1 | 0 | 3 | ... | 2 | 0 | 1 | 4 | 0 | 0 | 0 | 2 | 2 | 0 |
| 39 | 0 | 0 | 0 | 1 | 2 | 0 | 1 | 3 | 2 | 6 | ... | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 40 | 2 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 4 | ... | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 41 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 2 | 0 | 1 | ... | 0 | 2 | 1 | 1 | 0 | 2 | 1 | 0 | 2 | 0 |
| 42 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 3 | ... | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 2 | 2 | 2 |
| 43 | 1 | 2 | 0 | 4 | 0 | 3 | 1 | 1 | 3 | 16 | ... | 1 | 1 | 0 | 0 | 4 | 0 | 0 | 0 | 1 | 1 |
| 44 | 2 | 2 | 2 | 1 | 2 | 1 | 1 | 2 | 0 | 3 | ... | 1 | 1 | 1 | 0 | 4 | 0 | 0 | 0 | 1 | 4 |
| 45 | 3 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 2 | 0 | ... | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 |
| 46 | 6 | 2 | 0 | 2 | 0 | 1 | 0 | 2 | 0 | 1 | ... | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 5 |
| 47 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 3 | ... | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 48 | 3 | 2 | 0 | 1 | 2 | 1 | 2 | 1 | 1 | 6 | ... | 2 | 1 | 1 | 2 | 1 | 0 | 0 | 0 | 1 | 0 |
| 49 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | ... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 50 | 0 | 2 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 2 | ... | 2 | 4 | 2 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 51 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 0 | 2 | 3 | ... | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |