```
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import string
from string import punctuation
import os
from os import listdir
from collections import Counter
from tensorflow.keras.preprocessing.text import Tokenizer
from nltk.tokenize import word_tokenize
```

```
def file_to_string(filename):
 '''Opens the input text file and
 returns a string of all its text.'''
 file = open(filename, 'r')
 text = file.read()
 file.close()
 text = text.replace('\n', ' ')
 text = text.replace('  ', ' ')
 return text
```

```
cd ..
```

```
/Users/kamilapalys/Desktop/school/data450/capstone
```

```
pwd
```

```
'/Users/kamilapalys/Desktop/school/data450/capstone'
```

```
# example of using the function

filepath = 'data/text/cnn_trump.txt'
test_txt = file_to_string(filepath)
test_txt
```

```
"Donald Trump faces more than 30 counts related to business fraud in an indictment from a Ma
the first time in American history that a current or former president has faced criminal cha
```

which has never seen one of its ex-leaders confronted with criminal charges, let alone while
into uncharted waters. Trump released a statement in response to the indictment claiming it
united and strong - will first defeat Alvin Bragg, and then we will defeat Joe Biden, and we
or more - away. "Is this a shock today? Hell yes," the person said, speaking on a condition
as well as his 2024 GOP rivals - have condemned the Manhattan district attorney's office ove

```
def clean_text(text):
    '''Takes in a string of text and cleans it by converting
    to lowercase, removing punctuation, removing stopwords,
    and stemming. Returns the new string.'''
    ps = PorterStemmer()
    # create list of stopwords
    stopwords_list = stopwords.words('english')
    # make the text lowercase
    text = text.lower()
    # convert to ascii characters
    text = text.encode("ascii", "ignore").decode()
    for chr in text:
        # only keep characters in the string that are not punctuation symbols
        if chr in string.punctuation:
            text = text.replace(chr, ' ')
    text = text.replace('  ', ' ')
    # stem the tokens within the text
    tokens = text.split(" ")
    stemmed = []
    for token in tokens:
        stemmed_word = ps.stem(token)
        # only include new token in the cleaned list if not a stopword
        if stemmed_word not in stopwords_list:
            stemmed.append(stemmed_word)
    cleaned_text = " ".join(stemmed)
    return cleaned_text
```

```
cleaned_text = clean_text(test_txt)
cleaned_text
```

```
'donald trump face 30 count relat busi fraud indict manhattan grand juri accord two sourc fa
```

```
len(cleaned_text.split(" "))
```

711

```
# looping through all text files to apply preprocessing functions

file_list = []
article_docs = []
dir = os.listdir('data/text/')
dir.sort()
for filename in dir:
    filepath = os.path.join(directory, filename)
    file_list.append(f"{filepath}")
    if filename.split(".")[-1] == "txt":
        article_string = file_to_string(filepath)
        new_string = clean_text(article_string)
        article_docs.append(new_string)

#article_docs

t = Tokenizer()
t.fit_on_texts(article_docs)
print(t)
encoded_docs = t.texts_to_matrix(article_docs, mode='binary')
words = [x for x in t.word_index.keys()]
binary_df = pd.DataFrame(data = encoded_docs[:, 1:], columns=words)
binary_df

# convert the list of article strings into a dataframe
```

```
<keras.src.preprocessing.text.Tokenizer object at 0x178d7e6d0>
```

	said	wa	hi	ha	trump	thi	biden	offici	mr	u	...	messr	overlap	vs	convert	marku
0	1.0	1.0	1.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
1	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
2	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
3	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
4	1.0	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

	said	wa	hi	ha	trump	thi	biden	offici	mr	u	...	messr	overlap	vs	convert	marku
67	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
68	1.0	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0	0.0
69	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0
70	1.0	1.0	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0
71	1.0	1.0	1.0	1.0	1.0	1.0	0.0	0.0	1.0	1.0	...	1.0	1.0	1.0	1.0	1.0

```
doc1 = 'start engine'
doc2 = 'start your engine'
doc3 = 'whenever youre ready to start'
docs = [doc1, doc2, doc3]
```

```
# takes all docs and makes them into a dataframe
# make sure that docs is just a list of all the tokens from all articles
# first make one list of tokens for each article and then add all those lists up
# mode is one of "binary", "count", "tfidf", "freq"
t = Tokenizer()

t.fit_on_texts(docs)
print(t)
encoded_docs = t.texts_to_matrix(docs, mode='binary')
#print(encoded_docs)
words = [x for x in t.word_index.keys()]
bow = pd.DataFrame(data = encoded_docs[:, 1:], columns=words)
bow['source'] = ['cnn']*2 + ['abc']
bow
```

```
<keras.src.preprocessing.text.Tokenizer object at 0x178a732b0>
```

	start	engine	your	whenever	youre	ready	to	source
0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	cnn
1	1.0	1.0	1.0	0.0	0.0	0.0	0.0	cnn
2	1.0	0.0	0.0	1.0	1.0	1.0	1.0	abc

```
t = Tokenizer()

docs = [cleaned_text]
t.fit_on_texts(docs)
```

```
print(t)
encoded_docs = t.texts_to_matrix(docs, mode='binary')
#print(encoded_docs)
words = [x for x in t.word_index.keys()]
bow = pd.DataFrame(data = encoded_docs[:, 1:], columns=words)
bow
```

```
<keras.src.preprocessing.text.Tokenizer object at 0x178c5de20>
```

	trump	indict	presid	ha	hi	former	wa	offic	charg	said	...	420	origin	liabil	reward
0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	1.0	1.0	1.0	1.0

```
from sklearn.feature_extraction.text import CountVectorizer

data = {'text': docs, 'source': ['cnn', 'abc', 'nbc']}
df = pd.DataFrame(data)

vectorizer = CountVectorizer()
bow = vectorizer.fit_transform(df['text'])
print(bow)

count_array = bow.toarray()
features = vectorizer.get_feature_names()
print(count_array)
print(features)
#df = pd.DataFrame(data=count_array, columns=features)
```

```
AttributeError: 'list' object has no attribute 'lower'
```