```
import pandas as pd
import numpy as np
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

```
# initialize a dictionary to be able to find the original word from the stemmed word
stemmed_dict = {}

# how to later access the key by the value
#value = {i for i in dic if dic[i]=="B"}
#print("key by value:",value
```

```
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()
    text = text.replace('-', ' ')
    # 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[:-2]:
        # only include new token in the cleaned list if not a stopword
        if token not in stopwords_list:
            stemmed_word = ps.stem(token)
            stemmed.append(stemmed_word)
            if token not in stemmed_dict:
                stemmed_dict[token] = stemmed_word
    stemmed.append(tokens[-2])
```

```
        stemmed.append(tokens[-1])
        cleaned_text = " ".join(stemmed)
        return cleaned_text
```

```
text='-'

text = text.encode("ascii", "ignore").decode()

text
```

```
''
```

```
article_string = file_to_string('data/text/abc_hamas.txt')
new_string = clean_text(article_string)
new_string
```

```
'hundr peopl israel report dead thousand injur hama milit fire rocket gaza launch ground incu
```

```
# 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)

# convert the list of article strings into a binary-value dataframe
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)
# List of conditions
```

```
source_conditions = [
      binary_df['abcarticle'] == 1
    , binary_df['bbcarticle'] == 1
    , binary_df['cnnarticle'] == 1
    , binary_df['foxarticle'] == 1
    , binary_df['nbcarticle'] == 1
    , binary_df['nyparticle'] == 1
    , binary_df['nytarticle'] == 1
    , binary_df['wparticle'] == 1
    , binary_df['wsjarticle'] == 1
]

# List of values to return
source_choices  = [
      "ABC News"
    , "BBC"
    , "CNN"
    , "Fox News"
    , "NBC News"
    , "New York Post"
    , "The New York Times"
    , "The Washington Post"
    , "The Wall Street Journal"
]

# List of conditions
topic_conditions = [
      binary_df['affirmativearticle'] == 1
    , binary_df['balloonarticle'] == 1
    , binary_df['bidenarticle'] == 1
    , binary_df['hamasarticle'] == 1
    , binary_df['pentagonarticle'] == 1
    , binary_df['santosarticle'] == 1
    , binary_df['tanksarticle'] == 1
    , binary_df['trumparticle'] == 1
]
# List of values to return
topic_choices  = [
      "Supreme Court Ruling on Affirmative Action"
    , "Chinese Surveillance Balloon"
    , "Biden's Low Approval Rates in Polls"
```

```
    , "The Deadliest Attack by Hamas"
    , "Pentagon Documents Leak"
    , "George Santos' Expulsion from Congress"
    , "U.S. and Germany Send Tanks to Ukraine"
    , "Trump's Indictment"
]
# create a new source column
binary_df["article_source"] = np.select(source_conditions, source_choices, "ERROR")

# create a new topic column
binary_df["article_topic"] = np.select(topic_conditions, topic_choices, "ERROR")

binary_df.head()
```

```
<keras.src.preprocessing.text.Tokenizer object at 0x177a47070>
```

	said	trump	biden	offici	mr	u	israel	presid	tank	hous	...	messr	overlap	vs	convert
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
1	1.0	0.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0
2	1.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0
3	1.0	0.0	1.0	1.0	0.0	1.0	1.0	1.0	0.0	1.0	...	0.0	0.0	0.0	0.0
4	1.0	0.0	0.0	1.0	0.0	1.0	1.0	1.0	1.0	0.0	...	0.0	0.0	0.0	0.0

```
# check what word a given stemmed word represents
value = {i for i in stemmed_dict if stemmed_dict[i]=="marku"}
print("key by value:",value)
```

```
key by value: {'markus'}
```

```
encoded_docs_freq = t.texts_to_matrix(article_docs, mode='count')
freq_df = pd.DataFrame(data = encoded_docs_freq[:, 1:], columns=words)
# List of conditions
source_conditions = [
      freq_df['abcarticle'] == 1
    , freq_df['bbcarticle'] == 1
    , freq_df['cnnarticle'] == 1
    , freq_df['foxarticle'] == 1
```

```
    , freq_df['nbcarticle'] == 1
    , freq_df['nyparticle'] == 1
    , freq_df['nytarticle'] == 1
    , freq_df['wparticle'] == 1
    , freq_df['wsjarticle'] == 1
]

# List of values to return
source_choices  = [
      "ABC News"
    , "BBC"
    , "CNN"
    , "Fox News"
    , "NBC News"
    , "New York Post"
    , "The New York Times"
    , "The Washington Post"
    , "The Wall Street Journal"
]

# List of conditions
topic_conditions = [
      freq_df['affirmativearticle'] == 1
    , freq_df['balloonarticle'] == 1
    , freq_df['bidenarticle'] == 1
    , freq_df['hamasarticle'] == 1
    , freq_df['pentagonarticle'] == 1
    , freq_df['santosarticle'] == 1
    , freq_df['tanksarticle'] == 1
    , freq_df['trumparticle'] == 1
]
# List of values to return
topic_choices  = [
      "Supreme Court Ruling on Affirmative Action"
    , "Chinese Surveillance Balloon"
    , "Biden's Low Approval Rates in Polls"
    , "The Deadliest Attack by Hamas"
    , "Pentagon Documents Leak"
    , "George Santos' Expulsion from Congress"
    , "U.S. and Germany Send Tanks to Ukraine"
    , "Trump's Indictment"
```

```
]
# create a new source column
freq_df["article_source"] = np.select(source_conditions, source_choices, "ERROR")

# create a new topic column
freq_df["article_topic"] = np.select(topic_conditions, topic_choices, "ERROR")

freq_df.head()
```

	said	trump	biden	offici	mr	u	israel	presid	tank	hous	...	messr	overlap	vs	conver
0	5.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1	30.0	0.0	5.0	28.0	0.0	15.0	0.0	4.0	0.0	3.0	...	0.0	0.0	0.0	0.0
2	9.0	17.0	27.0	0.0	0.0	0.0	0.0	8.0	0.0	4.0	...	0.0	0.0	0.0	0.0
3	17.0	0.0	3.0	6.0	0.0	10.0	28.0	2.0	0.0	1.0	...	0.0	0.0	0.0	0.0
4	9.0	0.0	0.0	5.0	0.0	20.0	2.0	2.0	3.0	0.0	...	0.0	0.0	0.0	0.0

```
# create dataframe with tf-idf values

encoded_docs_tfidf = t.texts_to_matrix(article_docs, mode='tfidf')
tfidf_df = pd.DataFrame(data = encoded_docs_tfidf[:, 1:], columns=words)
# List of conditions
source_conditions = [
      tfidf_df['abcarticle'] != 0
    , tfidf_df['bbcarticle'] != 0
    , tfidf_df['cnnarticle'] != 0
    , tfidf_df['foxarticle'] != 0
    , tfidf_df['nbcarticle'] != 0
    , tfidf_df['nyparticle'] != 0
    , tfidf_df['nytarticle'] != 0
    , tfidf_df['wparticle'] != 0
    , tfidf_df['wsjarticle'] != 0
]

# List of values to return
source_choices  = [
      "ABC News"
    , "BBC"
    , "CNN"
    , "Fox News"
    , "NBC News"
```

```
    , "New York Post"
    , "The New York Times"
    , "The Washington Post"
    , "The Wall Street Journal"
]

# List of conditions
topic_conditions = [
      tfidf_df['affirmativearticle'] != 0
    , tfidf_df['balloonarticle'] != 0
    , tfidf_df['bidenarticle'] != 0
    , tfidf_df['hamasarticle'] != 0
    , tfidf_df['pentagonarticle'] != 0
    , tfidf_df['santosarticle'] != 0
    , tfidf_df['tanksarticle'] != 0
    , tfidf_df['trumparticle'] != 0
]
# List of values to return
topic_choices  = [
      "Supreme Court Ruling on Affirmative Action"
    , "Chinese Surveillance Balloon"
    , "Biden's Low Approval Rates in Polls"
    , "The Deadliest Attack by Hamas"
    , "Pentagon Documents Leak"
    , "George Santos' Expulsion from Congress"
    , "U.S. and Germany Send Tanks to Ukraine"
    , "Trump's Indictment"
]
# create a new source column
tfidf_df["article_source"] = np.select(source_conditions, source_choices, "ERROR")

# create a new topic column
tfidf_df["article_topic"] = np.select(topic_conditions, topic_choices, "ERROR")

tfidf_df.head()
```

	said	trump	biden	offici	mr	u	israel	presid	tank	hous	..
0	1.827036	0.000000	0.000000	0.000000	0.0	2.313275	0.000000	0.000000	0.000000	0.000000	..
1	3.081563	0.000000	2.267700	4.139471	0.0	3.594586	0.000000	1.881491	0.000000	1.871958	..
2	2.238584	5.086179	3.733245	0.000000	0.0	0.000000	0.000000	2.428008	0.000000	2.128570	..
3	2.683881	0.000000	1.823774	2.667558	0.0	3.201528	6.446654	1.334974	0.000000	0.891998	..

	said	trump	biden	offici	mr	u	israel	presid	tank	hous	..
4	2.238584	0.000000	0.000000	2.493348	0.0	3.873465	2.519533	1.334974	3.689062	0.000000	..