NLP Homework2
Name: Chaithanya Chikkannaswamy

NLP HOMEWORK 2

1. Data Cleaning

I have extracted the data from the text column from covid19.csv that I generated in the previous assignment.
In first step, I'm tokenizing each sentence and word of the text column using nltk.word_tokenize()

```
In [4]: #tokenization
        tokenized = df.apply(lambda row: nltk.word_tokenize(row['text']), axis=1)
        print(tokenized[:20])
```

```
0     [Bengaluru, :, Isolation, wards, in, hospitals...
1     [The, government, making, sure, that, the, new...
2     [Apart, from, more, people, falling, sick, (, ...
3     [Asian, stock, markets, are, mostly, higher, o...
4     [Cash, flow, was, also, “, very, strong, ,, ”,...
5     [CityNews, catches, up, with, arriving, passen...
6     [Because, :, *, Coronavirus, has, no, vaccine,...
7     [The, Chinese, Super, League, (, CSL, ), has, ...
8     [The, outbreak, of, coronavirus, is, quickly, ...
9     [The, Department, of, Health-Cordillera, Admin...
10    [Farrukhabad, hostage, crisis, ends, after, 8,...
11    [News, A, Toronto, hospital, says, a, man, wit...
12    [The, new, coronavirus, has, been, declared, a...
13    [REUTERS, file, photo, of, German, share, pric...
14    [Cases, also, reported, in, Russia, ,, as, US,...
15    [Man, Utd, in, talks, over, Odion, Ighalo, sig...
16    [California, pitches, Kobe, ,, Gianna, Bryant,...
17    [(, Bloomberg, ), --, Asian, stocks, looked, s...
18    [(, Bloomberg, ), --, Want, to, receive, this,...
19    [Jan, 30, ,, 2020, /, 03:26, PM, EST, /, Updat...
dtype: object
```

Python code for tokenization

I have used nltk.pos_tag() to find the parts of speech for each word in the dataset.

```
In [5]: pos_tag = [nltk.pos_tag(sent) for sent in tokenized]

In [6]: print(pos_tag[:20])
```

```
[[('Bengaluru', 'NN'), (':', ':'), ('Isolation', 'NN'), ('wards', 'NNS'), ('in', 'IN'), ('hospitals', 'NNS'), ('acros
s', 'IN'), ('Karnataka', 'NNP'), ('and', 'CC'), ('helpline', 'NN'), ('to', 'TO'), ('take', 'VB'), ('calls', 'NNS'),
('on', 'IN'), ('coronavirus-related', 'JJ'), ('queries', 'NNS'), ('are', 'VBP'), ('ready', 'JJ'), ('to', 'TO'), ('pre
vent', 'VB'), ('any', 'DT'), ('further', 'JJ'), ('spread', 'NN'), ('of', 'IN'), ('the', 'DT'), ('virus', 'NN'), ('aft
er', 'IN'), ('the', 'DT'), ('first', 'JJ'), ('case', 'NN'), ('in', 'IN'), ('India', 'NNP'), ('was', 'VBD'), ('reporte
d', 'VBN'), ('from', 'IN'), ('Kerala', 'NNP'), ('yesterday', 'NN'), ('.', '.'), ('The', 'DT'), ('Chief', 'NNP'), ('Se
cretary', 'NNP'), ('of', 'IN'), ('the', 'DT'), ('state', 'NN'), ('government', 'NN'), ('on', 'IN'), ('Thursday', 'NN
P'), ('held', 'VBD'), ('a', 'DT'), ('meeting', 'NN'), ('with', 'IN'), ('the', 'DT'), ('Additional', 'NNP'), ('Chief',
'NNP'), ('Secretary', 'NNP'), ('(', '('), ('Health', 'NNP'), (')', ')'), (',', ','), ('Health', 'NNP'), ('Commissione
r', 'NNP'), (',', ','), ('Mission', 'NNP'), ('Director', 'NNP'), ('of', 'IN'), ('the', 'DT'), ('National', 'NNP'),
('Health', 'NNP'), ('Mission', 'NNP'), ('and', 'CC'), ('other', 'JJ'), ('health', 'NN'), ('department', 'NN'), ('offi
cials', 'NNS'), ('and', 'CC'), ('reviewed', 'VBD'), ('the', 'DT'), ('state', 'NN'), ('preparedness', 'NN'), ('to', 'T
O'), ('tackle', 'VB'), ('any', 'DT'), ('cases', 'NNS'), ('of', 'IN'), ('coronavirus', 'NN'), ('whenever', 'NN'), ('re
ported', 'VBD'), ('.', '.'), ('Rajiv', 'NNP'), ('Gandhi', 'NNP'), ('Institute', 'NNP'), ('of', 'IN'), ('Chest', 'NN
P'), ('Diseases', 'NNP'), ('(', '('), ('RGICD', 'NNP'), (')', ')'), ('with', 'IN'), ('15', 'CD'), ('beds', 'NNS'),
('and', 'CC'), ('Wenlock', 'NNP'), ('Hospital', 'NNP'), ('at', 'IN'), ('Mangaluru', 'NNP'), ('with', 'IN'), ('10', 'C
D'), ('beds', 'NNS'), ('have', 'VBP'), ('been', 'VBN'), ('selected', 'VBN'), ('for', 'IN'), ('the', 'DT'), ('treatmen
t', 'NN'), ('of', 'IN'), ('the', 'DT'), ('virus', 'NN'), ('.', '.'), ('All', 'DT'), ('district', 'NN'), ('hospitals',
'NNS'), ('will', 'MD'), ('have', 'VB'), ('five', 'CD'), ('beds', 'NNS'), ('isolated', 'VBN'), ('for', 'IN'), ('patien
```

Python code to find the POS for each word

SUID: 387781563

NLP Homework2
Name: Chaithanya Chikkannaswamy

After tokenizing and getting the parts of speech for each word, to find the adjective phrases, I defined the grammar as "ADJP: {<RB.?>+<JJ.?>}". To determine the adjectives , I have used nltk.RegexParser(grammar). The adjective phrases is defined the grammar.

The adjective phrase start with any character except the newline continues with one or more characters followed by Adjective Phrase and ends with a character.

After defining grammar, I created an empty list to append all the adjectives. I used **nltk.tree** library and its inbuilt functions subtree(), label(), in order to extract the sentences for frequency analysis.

The subtree() function is used to trace each words according to grammar definition. Then I used label function to look for label in a sentence. If the label contains grammar definition it will be append into the list.

```
In [47]: #Grammar to get the adjectives
grammar1 = "ADJP: {<RB.?>+<JJ.?>}"
adj_ph = nltk.RegexpParser(grammar1)

adj_parser = []
for sent in pos_tag:
    if len(sent) > 0:
        tree = adj_ph.parse(sent)
        for subtree in tree.subtrees():
            if subtree.label() == 'ADJP':
                adj_parser.append(subtree)
```

```
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

```
In [48]: print(adj_parser[:100])
```

```
[Tree('ADJP', [('as', 'RB'), ('bad', 'JJ')]), Tree('ADJP', [('more', 'RBR'), ('fundamental', 'JJ')]), Tree('ADJP',
[('less', 'RBR'), ('developed', 'JJ')]), Tree('ADJP', [('more', 'RBR'), ('dangerous', 'JJ')]), Tree('ADJP', [('less',
'RBR'), ('stable', 'JJ')]), Tree('ADJP', [('even', 'RB'), ('more', 'RBR'), ('virulent', 'JJ')]), Tree('ADJP', [('most
ly', 'RB'), ('higher', 'JJR')]), Tree('ADJP', [('also', 'RB'), ('higher', 'JJR')]), Tree('ADJP', [('also', 'RB'),
('“', 'JJ')]), Tree('ADJP', [('very', 'RB'), ('strong', 'JJ')]), Tree('ADJP', [('as', 'RB'), ('much', 'JJ')]), Tree
('ADJP', [('Here', 'RB'), ('’', 'JJ')]), Tree('ADJP', [('extremely', 'RB'), ('impressive', 'JJ')]), Tree('ADJP', [('a
lso', 'RB'), ('lower', 'JJR')]), Tree('ADJP', [('not', 'RB'), ('impressive', 'JJ')]), Tree('ADJP', [('incrementally',
'RB'), ('supportive', 'JJ')]), Tree('ADJP', [('quickly', 'RB'), ('unraveling', 'JJ')]), Tree('ADJP', [('possibly', 'R
```

Python code to define the adjective word grammar and get the tree of adjectives

2. Descriptive statistics of the data

Average length of sentence:
To find the length of average length of sentences, I have defined a function avg which calculates the average of the list by dividing the length of the given list to the number of words in the list.

SUID: 387781563

```
In [60]: def avg(list):
             items = len(list)
             count = 0
             for sent in list:
                 count+= len(sent)
             return (count/items)

         print(avg(tokenized))
```

```
618.582614109058
```

Python code to calculate the average length of sentences

Top50 Adjective phrases by frequency:

For the frequency analysis, I have defined a Freq() which takes the list of words as an argument. Here, we first convert the given tokens to lower case using lower() and then remove the alpha numeric characters in the data using isalpha(). In order to refine the data set for my analysis I have also removed the stop words using nltk.corpus.stopwords.words('english'). Here, I'm displaying the top 50 adjective phrases by frequency.

```
In [64]: from nltk import FreqDist
         def freq(tokens):
             alpha_tokens = [w for w in tokens if w.isalpha()]
             alpha_lower_tokens = [w.lower() for w in alpha_tokens]
             stopwords = nltk.corpus.stopwords.words('english')
             stop_alpha_lower_tokens = [w for w in alpha_lower_tokens if w not in stopwords]
             freqs = FreqDist(stop_alpha_lower_tokens)
             top_freq = freqs.most_common(50)
             for frequency in top_freq:
                 print(frequency)
         freq(adj_parser)
```

Python code to get the top50 adjective phrases by frequency

```
('much', 11828)
('still', 6478)
('also', 5820)
('many', 4835)
('important', 4743)
('far', 4673)
('likely', 4616)
('even', 4588)
('really', 4060)
('early', 3792)
('less', 3637)
('relatively', 3549)
('highly', 3512)
('last', 3442)
('good', 3269)
('low', 3067)
('late', 2989)
('extremely', 2917)
('due', 2741)
('high', 2425)
('quite', 2357)
('higher', 2355)
('potentially', 2059)
('largest', 2041)
('serious', 2027)
('difficult', 1974)
('particularly', 1939)
('increasingly', 1921)
('available', 1889)
('already', 1875)
('lower', 1792)
('pretty', 1762)
('clear', 1732)
('effective', 1644)
('back', 1629)
('little', 1569)
```

Top 50 adjective phrase frequency

Top 50 adjective words:

To find the top 50 adjective words, I have conducted the same steps for data processing as above and getting the top 50 adjective words

```
In [66]: from nltk import FreqDist
         def freq(tokens):
             alpha_tokens = [w for w in tokens if w.isalpha()]
             alpha_lower_tokens = [w.lower() for w in alpha_tokens]
             stopwords = nltk.corpus.stopwords.words('english')
             stop_alpha_lower_tokens = [w for w in alpha_lower_tokens if w not in stopwords]
             topwords = FreqDist(stop_alpha_lower_tokens)
             top50_adj_words = topwords.most_common(50)
             for word in top50_adj_words:
                 print(word[0])
         freq(adj_parser)
```

```
much
still
also
many
important
far
likely
even
really
early
less
relatively
highly
last
good
low
late
extremely
due
high
quite
higher
potentially
largest
serious
```

Python code to get the top 50 adjective words

Top 50 Nouns:

To get the top 50 nouns, I have first defined the grammar for nouns as “NN: {<NN.?>}”

After defining grammar, I created an empty list to append all the adjectives. I used **nltk.tree** library and its inbuilt functions subtree(), label(), in order to extract the sentences for frequency analysis.

The subtree() function is used to trace each words according to grammar definition. Then I used label function to look for label in a sentence. If the label contains grammar definition it will be append into the list.

For the analysis part, I have defined a Freq() which takes the list of words as an argument. Here, we first convert the given tokens to lower case using lower() and then remove the alpha numeric characters in the data using isalpha(). In order to refine the data set for my analysis I have also removed the stop words using nltk.corpus.stopwords.words('english'). Here, I’m displaying the top 50 noun words

NLP Homework2
Name: Chaithanya Chikkannaswamy

```
In [69]: from nltk import FreqDist
         def freq(tokens):
             alpha_tokens = [w for w in tokens if w.isalpha()]
             stopwords = nltk.corpus.stopwords.words('english')
             stop_alpha_tokens = [w for w in alpha_tokens if w not in stopwords]
             freqs = FreqDist(stop_alpha_tokens)
             top_freq = freqs.most_common(50)
             for top50_words in top_freq:
                 print(top50_words[0])
         freq(noun_parser)
```

```
Coronavirus
China
coronavirus
BEIJING
News
February
Home
World
people
New
Bloomberg
Feb
death
number
Health
cases
Hong
Singapore
Oil
US
SINGAPORE
WASHINGTON
government
https
TOKYO
outbreak
```

Python code to display to top 50 noun

Interpretation of results:

In this assignment, at first, I have tokenized the data in text column in the covid19.csv file generated in the previous assignment. I have tokenized the words and sentences and used the nltk pos_tag() to get the parts of speech of each words. I have calculated the length of the sentences and the have found the average length of the sentence by dividing the total length by otal number of sentences.

According to the frequency analysis, I have processed the data and have found the top 50 adjective phrases by frequency in the given data set. In this analysis, I got to know the top 50 adjective words used with the frequency such as “much” appears “1182” times, “still” appears “6478” times.

In the analysis part of finding top 50 adjective words, I realized the top 50 adjective words being used in the data and likewise for top 50 nouns.

SUID: 387781563

NLP Homework2
Name: Chaithanya Chikkannaswamy

Sentiment Analysis:

According to me, we can give each word a score based on pre-defined dictionary of words and score. Then we can calculate the average score of each sentence. We can then set a threshold limit and classify the sentence as a positive or negative sentence based on their average score (1 or 0). Later, we can count the positive sentences and negative sentences to find out the overall score of the text.

SUID: 387781563