PDS ASSIGNMENT

- **Muralikrishna Maanukonda**
- **16330766**

1) (20 points) The data file contains tweets that have been pulled from Twitter. In this dataset
use the text data in the “OriginalTweet” column and perform the following:
a) Convert the text corpus into tokens.
b) Perform stop word removal.
c) Count Word frequencies
d) Create word clouds.

This report's objective is to analyze a dataset of tweets that were downloaded from Twitter. The text data in the Original Tweet column underwent the following processes in particular.

1) Converting the text corpus into tokens

The first step is to tokenize text data, which requires breaking the text up into separate words or tokens. We can do this using the nltk library in Python. We first import the data into a Panda's data frame, and then we pre-process the dataset by using the word_tokenize function to tokenize each tweet in the preprocessed_data column. As a result, a new column called "tokens" is formed that contains a list of tokens for each tweet.

```
import pandas as pnd
import nltk
import re
nltk.download('punkt')
from nltk.tokenize import word_tokenize
dfs = pd.read_csv("Corona_NLP_test.csv", encoding='latin1')
dfs['preprocessed_data'] = dfs['OriginalTweet'].apply(lambda x: re.sub(r'http\S+|www\S+|@\S+|[^\w\s]+', '', x))
# Tokenize the text data
dfs['tokens'] = dfs['preprocessed_data'].apply(lambda x: word_tokenize(x.lower()))
print(dfs.head())
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
   UserName  ScreenName             Location     TweetAt  \
0         1       44953                  NYC  02-03-2020
1         2       44954          Seattle, WA  02-03-2020
2         3       44955                  NaN  02-03-2020
3         4       44956          Chicagoland  02-03-2020
4         5       44957  Melbourne, Victoria  03-03-2020

                                       OriginalTweet           Sentiment  \
0  TRENDING: New Yorkers encounter empty supermar...  Extremely Negative
1  When I couldn't find hand sanitizer at Fred Me...            Positive
2  Find out how you can protect yourself and love...  Extremely Positive
3  #Panic buying hits #NewYork City as anxious sh...            Negative
4  #toiletpaper #dunnypaper #coronavirus #coronav...             Neutral

                                  preprocessed_data  \
0  TRENDING New Yorkers encounter empty supermark...
1  When I couldnt find hand sanitizer at Fred Mey...
2  Find out how you can protect yourself and love...
3  Panic buying hits NewYork City as anxious shop...
4  toiletpaper dunnypaper coronavirus coronavirus...

                                              tokens
0  [trending, new, yorkers, encounter, empty, sup...
1  [when, i, couldnt, find, hand, sanitizer, at, ...
2  [find, out, how, you, can, protect, yourself, ...
3  [panic, buying, hits, newyork, city, as, anxio...
4  [toiletpaper, dunnypaper, coronavirus, coronav...
```

2) Perform stop word removal

To improve performance and reduce noise, stop words—common terms with little meaning—can be removed from text data. Stop words can be removed with the Python nltk module's assistance. Before using a lambda function to remove the stop words from each tweet in the "tokens" column, we first import them from the nltk corpus.

```
from nltk.corpus import stopwords as sw
nltk.download('sw')
stop_words_1 = set(sw.words('english'))
# Remove the stop words
dfs['tokens'] = dfs['tokens'].apply(lambda x: [word for word in x if word not in stop_words_1])
print(dfs.head())
```

```
   UserName  ScreenName             Location     TweetAt  \
0         1       44953                  NYC  02-03-2020
1         2       44954          Seattle, WA  02-03-2020
2         3       44955                  NaN  02-03-2020
3         4       44956          Chicagoland  02-03-2020
4         5       44957  Melbourne, Victoria  03-03-2020

                                       OriginalTweet           Sentiment  \
0  TRENDING: New Yorkers encounter empty supermar...  Extremely Negative
1  When I couldn't find hand sanitizer at Fred Me...            Positive
2  Find out how you can protect yourself and love...  Extremely Positive
3  #Panic buying hits #NewYork City as anxious sh...            Negative
4  #toiletpaper #dunnypaper #coronavirus #coronav...             Neutral

                                  preprocessed_data  \
0  TRENDING New Yorkers encounter empty supermark...
1  When I couldnt find hand sanitizer at Fred Mey...
2  Find out how you can protect yourself and love...
3  Panic buying hits NewYork City as anxious shop...
4  toiletpaper dunnypaper coronavirus coronavirus...

                                          tokens
0  [trending, new, yorkers, encounter, empty, sup...
1  [couldnt, find, hand, sanitizer, fred, meyer, ...
2          [find, protect, loved, ones, coronavirus]
3  [panic, buying, hits, newyork, city, anxious, ...
4  [toiletpaper, dunnypaper, coronavirus, coronav...
[nltk_data] Error loading sw: Package 'sw' not found in index
```

3) Count word frequencies

After tokenizing the text input and eliminating the stop words, we can now use the Python collections library to count the frequency of each word. After using a list comprehension to flatten the list of tokens, the Counter function is used to calculate the frequency of each word.

```
[13]
from collections import Counter
word_freq_1 = Counter([word for tokens in dfs["tokens"] for word in tokens])
print(word_freq_1.most_common(10))
```

```
[('covid_19', 1525), ('coronavirus', 1503), ('food', 1327), ('store', 1008), ('covid19', 962), ('grocery', 815), ('stock'
```

4) Create word clouds.

We may create word clouds to display the phrases that appear most frequently in the tweet data. We can utilize the Python wordcloud library for this.

```python
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# create word cloud
wordcloud_a = WordCloud(width=800, height=500, random_state=21, max_font_size=110, background_color='skyblue', max_words=100).generate_from_frequencies(word_freq)
plt.figure(figsize=(10, 7))
plt.imshow(wordcloud_a, interpolation="bilinear")
plt.axis('off')
plt.show()
```