TEXT MINING FINAL PRESENTATION

SEAN GRAHAM

INTRO

You are a data scientist in healthcare. Your manager gives you two Coronavirus tweets datasets. The training dataset contains in the file Corona_NLP_train.csv. The test dataset contains in the file Corona_NLP_test.csv. Both have the following columns:

- UserName
- ScreenName
- Location
- TweetAt
- OriginalTweet
- Sentiment

IMPORT NECESSARY ITEMS

In [1]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import pyLDAvis
import pyLDAvis.sklearn
import re
import scattertext as st
import seaborn as sns
import spacy

from scipy import spatial
from sklearn import preprocessing
from sklearn import svm
from sklearn.base import BaseEstimator
from sklearn.base import TransformerMixin
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import SGDClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import plot_confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from pprint import pprint
from spacy import displacy
from spacy.lang.en.stop_words import STOP_WORDS
from time import time

# Many unnecssary warnings appear when this program is run; I
# belive it's best to disable them
import warnings
warnings.filterwarnings("ignore")
```

IMPORT DATASETS

In [2]:
```
# The Train dataset has over 41,000 rows; to save time, I'll use the first 1,000
Train_Data = pd.read_csv('Corona_NLP_train.csv', encoding = 'latin-1').head(1000)
# The test dataset has nearly 3,800 rows; to save time, I'll use the first 100
Test_Data = pd.read_csv('Corona_NLP_test.csv').head(100)
```

DISPLAY TRAIN DATA

```
In [3]: Train_Data.head(10)
```

Out[3]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative

DISPLAY TEST DATA

```
In [4]: Test_Data.head(10)
```

Out[4]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive
2	3	44955	NaN	02-03-2020	Find out how you can protect yourself and love...	Extremely Positive
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral
6	7	44959	NaN	03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can□t stop #COVID19 without prot...	Neutral
8	9	44961	NaN	04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive

1. CONCATENATE THE FOLLOWING THREE COLUMNS INTO A NEW COLUMN OF TWEET_TEXTS:
- LOCATION - TWEETAT - ORIGINALTWEET

```
In [5]: # I felt it was best to remove the location "NaN" values before performing the concatenation
        Train_Data = Train_Data.fillna('')
        Test_Data = Test_Data.fillna('')

        Concatenation = ["Location", "TweetAt", "OriginalTweet"]
        Train_Data["Tweet_texts"] = Train_Data[Concatenation].apply(lambda row: ' '.join(row.values.astype(str)), axis=1)
        Test_Data["Tweet_texts"] = Test_Data[Concatenation].apply(lambda row: ' '.join(row.values.astype(str)), axis=1)
```

```
In [6]: Train_Data.head(10)
```

Out[6]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London 16-03-2020 @MeNyrbie @Phil_Gahan @Chris...
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK 16-03-2020 advice Talk to your neighbours f...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds 16-03-2020 Coronavirus Australia: Wo...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	16-03-2020 My food stock is not the only one ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	16-03-2020 Me, ready to go at supermarket dur...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÃs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 16-03-2020 As news o...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 16-03-2020 Cashier at gro...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria 16-03-2020 Was at the supermarket toda...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA 16-03-2020 Due to COVID-19 our...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT 16-03-2020 For corona prevent...

```
In [7]: Test_Data.head(10)
```

Out[7]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC 02-03-2020 TRENDING: New Yorkers encounter...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle, WA 02-03-2020 When I couldn't find ha...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	02-03-2020 Find out how you can protect yours...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland 02-03-2020 #Panic buying hits #New...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne, Victoria 03-03-2020 #toiletpaper #d...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles 03-03-2020 Do you remember the las...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	03-03-2020 Voting in the age of #coronavirus ...
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can't stop #COVID19 without prot...	Neutral	Geneva, Switzerland 03-03-2020 @DrTedros "We c...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	04-03-2020 HI TWITTER! I am a pharmacist. I s...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin, Ireland 04-03-2020 Anyone been in a su...

2. CLEAN AND PREPROCESS THE NEW COLUMN OF TWEET_TEXTS:
2.1 REMOVE THE DATE AND TIME IN THE TWEETS USING A REGULAR EXPRESSION.

In [8]:
```
for i in Train_Data["Tweet_texts"]:
    edited_text = re.sub(r'(\d{1,2}-)?(\s)?\d{1,2}(:\d{2})?(\s)?(am|pm|AM|PM)(\s)?([A-Z]{3})?', ' ', i)
    edited_text = re.sub(r'(\d{1,2}-)?(\s)?\d{1,2}:\d{2}(\s)?(am|pm|AM|PM)?(\s)?([A-Z]{3})?', ' ', edited_text)

    edited_text = re.sub(r'\d{1,2}(-|/)\d{1,2}((-|/)\d{2,4})?', ' ', edited_text)
    edited_text = re.sub(r'(\d{1,2})?(st|nd|rd|th)?\s?(Jan(uary)?|JAN(UARY)?|Feb(ruary)?|FEB(RUARY)?|Mar(ch)? \
                         |MAR(CH)?|Apr(il)?|APR(IL)?|May|MAY|Jun(e)?|JUN(E)?|Jul(y)?|JUL(Y)?|Aug(ust)?|AUG(UST)? \
                         |Sep(tember)?|SEP(TEMBER)?|Oct(ober)?|OCT(OBER)?|Nov(ember)?|NOV(EMBER)?|Dec(ember)? \
                         |DEC(EMBER)?)(\s)?(\d{1,2})?(st|nd|rd|th)?(,)?(\s)?(\d{2,4})?', ' ', edited_text)

    Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
Train_Data.head(10)
```

Out[8]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London @MeNyrbie @Phil_Gahan @Chrisitv https...
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK advice Talk to your neighbours family to ...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	My food stock is not the only one which is ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	Me, ready to go at supermarket during the #...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 As news of the reg...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier at grocery stor...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was at the supermarket today. Didn't...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA Due to COVID-19 our retail s...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention,we sh...

In [9]:
```
for i in Test_Data["Tweet_texts"]:
    edited_text = re.sub(r'(\d{1,2}-)?(\s)?\d{1,2}(:\d{2})?(\s)?(am|pm|AM|PM)(\s)?([A-Z]{3})?', ' ', i)
    edited_text = re.sub(r'(\d{1,2}-)?(\s)?\d{1,2}:\d{2}(\s)?(am|pm|AM|PM)?(\s)?([A-Z]{3})?', ' ', edited_text)

    edited_text = re.sub(r'\d{1,2}(-|/)\d{1,2}((-|/)\d{2,4})?', ' ', edited_text)
    edited_text = re.sub(r'(\d{1,2})?(st|nd|rd|th)?\s?(Jan(uary)?|JAN(UARY)?|Feb(ruary)?|FEB(RUARY)?|Mar(ch)? \
                         |MAR(CH)?|Apr(il)?|APR(IL)?|May|MAY|Jun(e)?|JUN(E)?|Jul(y)?|JUL(Y)?|Aug(ust)?|AUG(UST)? \
                         |Sep(tember)?|SEP(TEMBER)?|Oct(ober)?|OCT(OBER)?|Nov(ember)?|NOV(EMBER)?|Dec(ember)? \
                         |DEC(EMBER)?)(\s)?(\d{1,2})?(st|nd|rd|th)?(,)?(\s)?(\d{2,4})?', ' ', edited_text)

    Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
Test_Data.head(10)
```

Out[9]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING: New Yorkers encounter empty su...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle, WA When I couldn't find hand saniti...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and l...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland #Panic buying hits #NewYork City...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne, Victoria #toiletpaper #dunnypaper...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles Do you remember the last time yo...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting in the age of #coronavirus = hand sa...
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We canÛt stop #COVID19 without prot...	Neutral	Geneva, Switzerland @DrTedros "We canÛt stop...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	HI TWITTER! I am a pharmacist. I sell hand ...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin, Ireland Anyone been in a supermarket...

2.2 REMOVE THE HYPERLINK URL IN THE TWEETS USING A REGULAR EXPRESSION.

```
In [10]: for i in Train_Data["Tweet_texts"]:
             edited_text = re.sub(r'https://\S*', '', i)
             Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
         Train_Data.head(10)
```

Out[10]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London @MeNyrbie @Phil_Gahan @Chrisitv and ...
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK advice Talk to your neighbours family to ...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	My food stock is not the only one which is ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	Me, ready to go at supermarket during the #...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÃs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 As news of the reg...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier at grocery stor...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was at the supermarket today. Didn't...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA Due to COVID-19 our retail s...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention,we sh...


```
In [10]: for i in Train_Data["Tweet_texts"]:
             edited_text = re.sub(r'https://\S*', '', i)
             Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
         Train_Data.head(10)
```

Out[10]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London @MeNyrbie @Phil_Gahan @Chrisitv and ...
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK advice Talk to your neighbours family to ...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	My food stock is not the only one which is ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	Me, ready to go at supermarket during the #...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÃs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 As news of the reg...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier at grocery stor...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was at the supermarket today. Didn't...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA Due to COVID-19 our retail s...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention,we sh...

2.3 REMOVE THE TWITTER HASHTAGS (# BEFORE A RELEVANT KEYWORD OR PHRASE) IN THE TWEETS USING A REGULAR EXPRESSION.

In [12]:
```
for i in Train_Data["Tweet_texts"]:
    edited_text = re.sub(r'#\S*', '', i)
    Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
Train_Data.head(10)
```

Out[12]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London @MeNyrbie @Phil_Gahan @Chrisitv and ...
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK advice Talk to your neighbours family to ...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	My food stock is not the only one which is ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	Me, ready to go at supermarket during the ...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÃs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 As news of the reg...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier at grocery stor...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was at the supermarket today. Didn't...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA Due to COVID-19 our retail s...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention,we sh...

In [13]:
```
for i in Test_Data["Tweet_texts"]:
    edited_text = re.sub(r'#\S*', '', i)
    Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
Test_Data.head(10)
```

Out[13]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING: New Yorkers encounter empty su...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle, WA When I couldn't find hand saniti...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and l...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland buying hits City as anxious sh...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne, Victoria One week ...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles Do you remember the last time yo...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting in the age of = hand sanitizer ?
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can't stop #COVID19 without prot...	Neutral	Geneva, Switzerland @DrTedros "We can't stop...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	HI TWITTER! I am a pharmacist. I sell hand ...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin, Ireland Anyone been in a supermarket...

2.4 REMOVE THE USERNAMES (ALSO KNOWN AS THE HANDLE — BEGINS WITH THE "@" SYMBOL) USING A REGULAR EXPRESSION.

In [14]:
```
for i in Train_Data["Tweet_texts"]:
    edited_text = re.sub(r'@\S*', '', i)
    Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
Train_Data.head(10)
```

Out[14]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London and and
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	UK advice Talk to your neighbours family to ...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	My food stock is not the only one which is ...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	Me, ready to go at supermarket during the ...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	ÃT: 36.319708,-82.363649 As news of the reg...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier at grocery stor...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was at the supermarket today. Didn't...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, GA USA Due to COVID-19 our retail s...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention,we sh...

In [15]:
```
for i in Test_Data["Tweet_texts"]:
    edited_text = re.sub(r'@\S*', '', i)
    Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
Test_Data.head(10)
```

Out[15]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING: New Yorkers encounter empty su...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle, WA When I couldn't find hand saniti...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and l...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland buying hits City as anxious sh...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne, Victoria One week ...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles Do you remember the last time yo...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting in the age of = hand sanitizer ?
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can't stop #COVID19 without prot...	Neutral	Geneva, Switzerland "We can't stop without...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	HI TWITTER! I am a pharmacist. I sell hand ...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin, Ireland Anyone been in a supermarket...

2.5 REMOVE ALL WORDS CONTAINING AT MOST TWO CHARACTERS SUCH AS "A", "AN", "IN", "ON", ETC.

In [16]:
```
for i in Train_Data["Tweet_texts"]:
    edited_text = re.sub(r'\b[a-zA-Z]{1,2}\b', '', i)
    Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
Train_Data.head(10)
```

Out[16]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London and and
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	advice Talk your neighbours family exchan...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia: Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	food stock not the only one which empty....
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	, ready supermarket during the outbreak...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	Ã: 36.319708,-82.363649 news the regionÂ...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	35.926541,-78.753267 Cashier grocery store ...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was the supermarket today. Didn' bu...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta, USA Due COVID-19 our retail store...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGAR,GUJRAT For corona prevention, shou...

In [17]:
```
for i in Test_Data["Tweet_texts"]:
    edited_text = re.sub(r'\b[a-zA-Z]{1,2}\b', '', i)
    Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
Test_Data.head(10)
```

Out[17]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING: New Yorkers encounter empty su...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle, When couldn' find hand sanitizer ...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and l...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland buying hits City anxious shop...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne, Victoria One week ...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles you remember the last time you ...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting the age = hand sanitizer ?
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can t stop #COVID19 without prot...	Neutral	Geneva, Switzerland " can stop without pr...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	TWITTER! pharmacist. sell hand sanitiz...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin, Ireland Anyone been supermarket ov...

2.6 REMOVE ALL SPECIAL CHARACTERS, PUNCTUATION USING A REGULAR EXPRESSION.

In [18]:
```
for i in Train_Data["Tweet_texts"]:
    edited_text = re.sub(r'[^\w\s]', '', i)
    Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]
Train_Data.head(10)
```

Out[18]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London and and
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	advice Talk your neighbours family exchan...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia Woolworths ...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	food stock not the only one which empty\...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	ready supermarket during the outbreak\...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	Ã 3631970882363649 news the regionÂ first ...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	3592654178753267 Cashier grocery store was ...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was the supermarket today Didn buy ...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta USA Due COVID19 our retail store a...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGARGUJRAT For corona prevention should...

In [19]:
```
for i in Test_Data["Tweet_texts"]:
    edited_text = re.sub(r'[^\w\s]', '', i)
    Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
Test_Data.head(10)
```

Out[19]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING New Yorkers encounter empty sup...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle When couldn find hand sanitizer F...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and l...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland buying hits City anxious shop...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne Victoria One week e...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles you remember the last time you ...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting the age hand sanitizer
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros “We can't stop #COVID19 without prot...	Neutral	Geneva Switzerland can stop without prote...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	TWITTER pharmacist sell hand sanitizer...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin Ireland Anyone been supermarket ove...

2.7 REMOVE ALL ROWS WITHOUT ANY TEXT LEFT (EMPTY TEXT) IN THE COLUMN OF TWEET_TEXTS.

```
In [20]: # Before doing this problem, I belive it is wise to remove extra spaces from the Tweet_text
         # entries. This includes two or more consecutive spaces within the entries and extra
         # whitespaces at the beginnings and ends of the entires.
         for i in Train_Data["Tweet_texts"]:
             edited_text = re.sub(r'\s{2,100}', ' ', i)
             edited_text = edited_text.strip()
             Train_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Train_Data["Tweet_texts"]]

         for i in Test_Data["Tweet_texts"]:
             edited_text = re.sub(r'\s{2,100}', ' ', i)
             edited_text = edited_text.strip()
             Test_Data["Tweet_texts"] = [x.replace(i, edited_text) for x in Test_Data["Tweet_texts"]]
```

```
In [21]: Train_Data = Train_Data[Train_Data.Tweet_texts != '']
         Train_Data.head(10)
```

Out[21]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	London and and
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	advice Talk your neighbours family exchange ph...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	Vagabonds Coronavirus Australia Woolworths giv...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	food stock not the only one which empty PLEASE...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	ready supermarket during the outbreak Not beca...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	Ã 3631970882363649 news the regionÂ first conf...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	3592654178753267 Cashier grocery store was sha...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	Austria Was the supermarket today Didn buy toi...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	Atlanta USA Due COVID19 our retail store and c...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	BHAVNAGARGUJRAT For corona prevention should s...

```
In [22]: Test_Data = Test_Data[Test_Data.Tweet_texts != '']
         Test_Data.head(10)
```

Out[22]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	NYC TRENDING New Yorkers encounter empty super...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	Seattle When couldn find hand sanitizer Fred M...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	Find out how you can protect yourself and love...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	Chicagoland buying hits City anxious shoppers ...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	Melbourne Victoria One week everyone buying ba...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	Los Angeles you remember the last time you pai...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	Voting the age hand sanitizer
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can't stop #COVID19 without prot...	Neutral	Geneva Switzerland can stop without protecting...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	TWITTER pharmacist sell hand sanitizer for liv...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	Dublin Ireland Anyone been supermarket over th...

2.8 PERFORM THE PART OF SPEECH TAGGING FOR THE TEXTS.

```
In [23]: nlp = spacy.load("en_core_web_sm")
```

```
In [24]: for i in Train_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             for token in nlp_i:
                 print(token.text, token.lemma_, token.pos_,
                       token.tag_, token.dep_, token.shape_,
                       token.is_alpha, token.is_stop)
             print('')
```

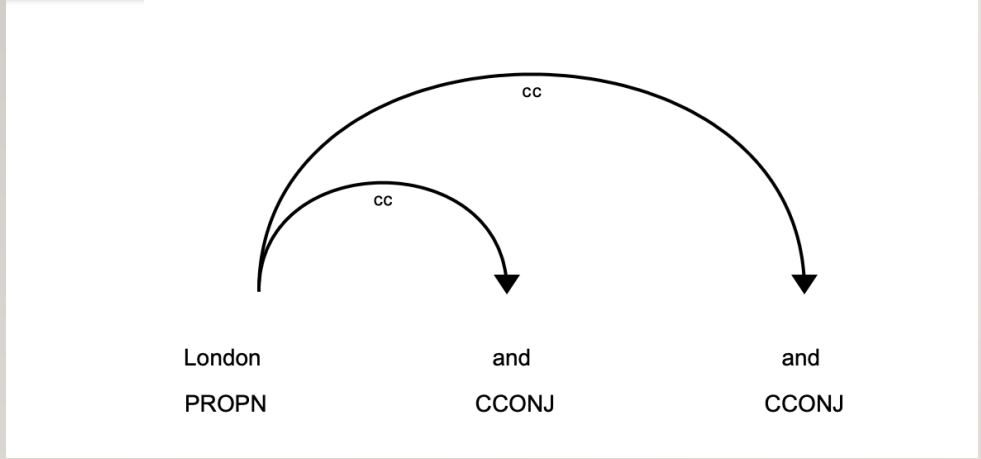
```
London London PROPN NNP ROOT Xxxxx True False
and and CCONJ CC cc xxx True True
and and CCONJ CC cc xxx True True

advice advice NOUN NN nsubj xxxx True False
Talk talk VERB VBP ROOT Xxxx True False
your your PRON PRP$ poss xxxx True True
neighbours neighbour NOUN NNS compound xxxx True False
family family NOUN NN compound xxxx True False
exchange exchange NOUN NN compound xxxx True False
phone phone NOUN NN compound xxxx True False
numbers number NOUN NNS nsubj xxxx True False
create create VERB VBP ccomp xxxx True False
contact contact NOUN NN compound xxxx True False
list list NOUN NN dobj xxxx True False
with with ADP IN prep xxxx True True
phone phone NOUN NN compound xxxx True False
numbers number NOUN NNS pobj xxxx True False
neighbours neighbour NOUN NNS intj xxxx True False
```

```
In [25]: for i in Test_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             for token in nlp_i:
                 print(token.text, token.lemma_, token.pos_,
                       token.tag_, token.dep_, token.shape_,
                       token.is_alpha, token.is_stop)
             print('')
```

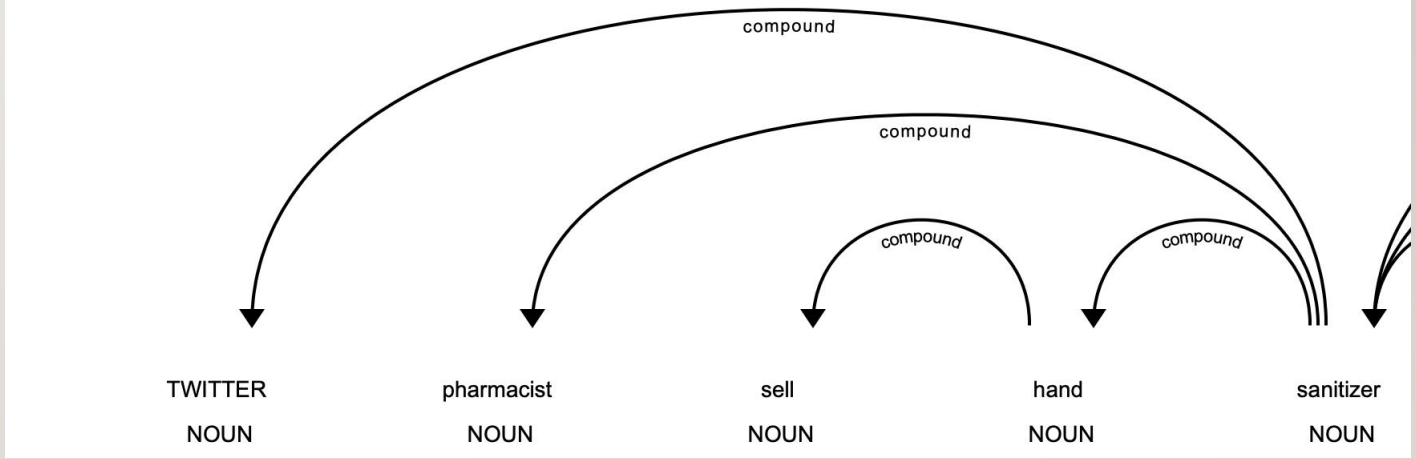
```
NYC NYC PROPN NNP nsubj XXX True False
TRENDING trending VERB VBP ROOT XXXX True False
New New PROPN NNP compound Xxx True False
Yorkers Yorkers PROPN NNPS dobj Xxxxx True False
encounter encounter VERB VBP dep xxxx True False
empty empty ADJ JJ amod xxxx True True
supermarket supermarket NOUN NN compound xxxx True False
shelves shelf NOUN NNS dobj xxxx True False
pictured picture VERB VBD acl xxxx True False
Wegmans Wegmans PROPN NNP compound Xxxxx True False
Brooklyn Brooklyn PROPN NNP dobj Xxxxx True False
soldout soldout VERB VBP prep xxxx True False
online online ADJ JJ amod xxxx True False
grocers grocer NOUN NNS dobj xxxx True False
FoodKick FoodKick PROPN NNP compound XxxxXxxx True False
MaxDelivery MaxDelivery PROPN NNP compound XxxXxxxx True False
shoppers shopper NOUN NNS compound xxxx True False
stock stock NOUN NN dobj xxxx True False
```

2.9 VISUALIZE THE DEPENDENCY PARSER.

In [26]:
```
# Displaying all the plots created significant lag, so I'm
# only displaying 10
for i in Train_Data["Tweet_texts"].head(10):
    nlp_i = nlp(i)
    displacy.render(nlp_i, style = "dep", jupyter = True)
```



In [27]:
```
# Displaying all the plots created significant lag, so I'm only displaying 10
for i in Test_Data["Tweet_texts"].head(10):
    nlp_i = nlp(i)
    displacy.render(nlp_i, style = "dep", jupyter = True)
```



2.10 PERFORM THE NAMED ENTITIES RECOGNITION FOR THE TEXTS.

```
In [28]: for i in Train_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             for ent in nlp_i.ents:
                 print(ent.text, ent.start_char,
                       ent.end_char, ent.label_)
             print('')
```

```
London 0 6 GPE


hours 84 89 TIME
COVID19 95 102 PERSON



3631970882363649 2 18 CARDINAL
first 36 41 ORDINAL
Sullivan County 74 89 GPE
last week 90 99 DATE

Cashier 17 24 ORG
Civics 96 102 ORG

Austria 0 7 PERSON
Didn 34 38 ORG
```

```
In [29]: for i in Test_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             for ent in nlp_i.ents:
                 print(ent.text, ent.start_char,
                       ent.end_char, ent.label_)
             print('')
```

```
New Yorkers 13 24 NORP
Wegmans 70 77 NORP
Brooklyn 78 86 GPE
FoodKick MaxDelivery 110 130 PRODUCT

Seattle 0 7 GPE
Fred Meyer 40 50 PERSON
11497 62 67 DATE
2 72 73 CARDINAL


Chicagoland 0 11 GPE
City 24 28 ORG
30s 93 96 DATE
1st 105 108 ORDINAL

Melbourne 0 9 GPE
One week 19 27 DATE
```

2.11 VISUALIZE THE GEOLOCATION, MONEY, AND QUANTITY IN THE TEXTS.

```
In [30]: # There is no NER type "GEOLOCATION", so I split it into
         # "GPE" and "LOC"
         options = {"ents": ['GPE', 'LOC', 'MONEY', 'QUANTITY'],
                    "colors": {'GPE':'#c0c623', 'LOC':'#c0c623',
                     'MONEY':'#caffcf', 'QUANTITY':'#caedff'}}
```

```
In [31]: for i in Train_Data["Tweet_texts"]:
             nlp_i = nlp(i)
             displacy.render(nlp_i, style = "ent", jupyter = True, options = options)
```

London GPE and and

advice Talk your neighbours family exchange phone numbers create contact list with phone numbers neig

shopping accounts poss adequate supplies regular meds but not over order

Vagabonds Coronavirus Australia Woolworths give elderly disabled dedicated shopping hours amid COVII

food stock not the only one which empty PLEASE don panic THERE WILL ENOUGH FOOD FOR EVERYO

stay safe

ready supermarket during the outbreak Not because paranoid but because food stock litteraly empty The

shortage

```
In [32]: for i in Test_Data["Tweet_texts"]:
             nlp_i = nlp(i)
             displacy.render(nlp_i, style = "ent", jupyter = True, options = options)
```

NYC TRENDING New Yorkers encounter empty supermarket shelves pictured Wegmans Brooklyn GPE s

shoppers stock

Seattle GPE When couldn find hand sanitizer Fred Meyer turned But 11497 for 2 pack PurellCheck out h

Find out how you can protect yourself and loved ones from

Chicagoland GPE buying hits City anxious shoppers stock foodampmedical supplies after worker her 30

Melbourne GPE Victoria One week everyone buying baby milk powder the next everyone buying toilet pa

Los Angeles GPE you remember the last time you paid 299 gallon QUANTITY for regular gas Los An

look how the impacting prices

3. EXTRACT ALL THE TOKENS FROM THE TWEET_TEXTS.

```
In [33]: for i in Train_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             tokens = [token.text for token in nlp_i]
             print(tokens)
             print('')
```

```
['London', 'and', 'and']

['advice', 'Talk', 'your', 'neighbours', 'family', 'exchange', 'phone', 'numbers', 'c
th', 'phone', 'numbers', 'neighbours', 'schools', 'employer', 'chemist', 'set', 'onli
poss', 'adequate', 'supplies', 'regular', 'meds', 'but', 'not', 'over', 'order']

['Vagabonds', 'Coronavirus', 'Australia', 'Woolworths', 'give', 'elderly', 'disabled'
urs', 'amid', 'COVID19', 'outbreak']

['food', 'stock', 'not', 'the', 'only', 'one', 'which', 'empty', 'PLEASE', 'don', 'pa
', 'FOOD', 'FOR', 'EVERYONE', 'you', 'not', 'take', 'more', 'than', 'you', 'need', 'S

['ready', 'supermarket', 'during', 'the', 'outbreak', 'Not', 'because', 'paranoid', '
k', 'litteraly', 'empty', 'The', 'serious', 'thing', 'but', 'please', 'don', 'panic',

['Ã', '3631970882363649', 'news', 'the', 'regionÂ', 'first', 'confirmed', 'COVID19',
an', 'County', 'last', 'week', 'people', 'flocked', 'area', 'stores', 'purchase', 'cl
sanitizer', 'food', 'toilet', 'paper', 'and', 'other', 'goods', 'reports']
```

```
In [34]: for i in Test_Data["Tweet_texts"]:
             nlp_i = nlp(i)

             tokens = [token.text for token in nlp_i]
             print(tokens)
             print('')
```

```
['NYC', 'TRENDING', 'New', 'Yorkers', 'encounter', 'empty', 'supermarket', 'shelves',
lyn', 'soldout', 'online', 'grocers', 'FoodKick', 'MaxDelivery', 'shoppers', 'stock']

['Seattle', 'When', 'couldn', 'find', 'hand', 'sanitizer', 'Fred', 'Meyer', 'turned',
pack', 'PurellCheck', 'out', 'how', 'concerns', 'are', 'driving', 'prices']

['Find', 'out', 'how', 'you', 'can', 'protect', 'yourself', 'and', 'loved', 'ones', '

['Chicagoland', 'buying', 'hits', 'City', 'anxious', 'shoppers', 'stock', 'foodampmed
orker', 'her', '30s', 'becomes', '1st', 'confirmed', 'patient', 'staged', 'event']

['Melbourne', 'Victoria', 'One', 'week', 'everyone', 'buying', 'baby', 'milk', 'powde
'buying', 'toilet', 'paper']

['Los', 'Angeles', 'you', 'remember', 'the', 'last', 'time', 'you', 'paid', '299', 'g
', 'Los', 'AngelesPrices', 'the', 'pump', 'are', 'going', 'down', 'look', 'how', 'the

['Voting', 'the', 'age', 'hand', 'sanitizer']
```

4. REPLACE THE TWEET_TEXTS COLUMN WITH THE STRING ONLY CONTAINING THE LEMMAS OF ALL THE TOKENS.

```
In [35]: def lemmatize(sentence):
             text = [word.lemma_ for word in nlp(sentence)]
             return ' '.join(text).strip().lower()
```

```
In [36]: Train_Data["Tweet_texts"] = Train_Data["Tweet_texts"].apply(lemmatize)
         Train_Data.head(10)
```

Out[36]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral	london and and
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive	advice talk your neighbour family exchange pho...
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive	vagabonds coronavirus australia woolworths giv...
3	3802	48754		16-03-2020	My food stock is not the only one which is emp...	Positive	food stock not the only one which empty please...
4	3803	48755		16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative	ready supermarket during the outbreak not beca...
5	3804	48756	ÃT: 36.319708,-82.363649	16-03-2020	As news of the regionÂs first confirmed COVID...	Positive	ã 3631970882363649 news the regionâ first conf...
6	3805	48757	35.926541,-78.753267	16-03-2020	Cashier at grocery store was sharing his insig...	Positive	3592654178753267 cashier grocery store be shar...
7	3806	48758	Austria	16-03-2020	Was at the supermarket today. Didn't buy toile...	Neutral	austria be the supermarket today didn buy toil...
8	3807	48759	Atlanta, GA USA	16-03-2020	Due to COVID-19 our retail store and classroom...	Positive	atlanta usa due covid19 our retail store and c...
9	3808	48760	BHAVNAGAR,GUJRAT	16-03-2020	For corona prevention,we should stop to buy th...	Negative	bhavnagargujrat for corona prevention should s...

```
In [37]: Test_Data["Tweet_texts"] = Test_Data["Tweet_texts"].apply(lemmatize)
         Test_Data.head(10)
```

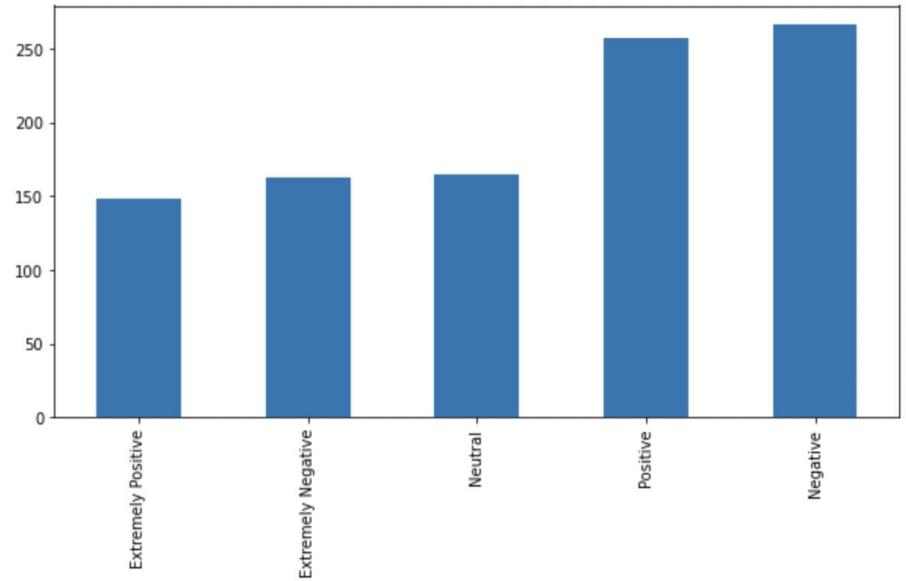
Out[37]:

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment	Tweet_texts
0	1	44953	NYC	02-03-2020	TRENDING: New Yorkers encounter empty supermar...	Extremely Negative	nyc trending new yorkers encounter empty super...
1	2	44954	Seattle, WA	02-03-2020	When I couldn't find hand sanitizer at Fred Me...	Positive	seattle when couldn find hand sanitizer fred m...
2	3	44955		02-03-2020	Find out how you can protect yourself and love...	Extremely Positive	find out how you can protect yourself and love...
3	4	44956	Chicagoland	02-03-2020	#Panic buying hits #NewYork City as anxious sh...	Negative	chicagoland buying hit city anxious shopper st...
4	5	44957	Melbourne, Victoria	03-03-2020	#toiletpaper #dunnypaper #coronavirus #coronav...	Neutral	melbourne victoria one week everyone buy baby ...
5	6	44958	Los Angeles	03-03-2020	Do you remember the last time you paid $2.99 a...	Neutral	los angeles you remember the last time you pay...
6	7	44959		03-03-2020	Voting in the age of #coronavirus = hand sanit...	Positive	vote the age hand sanitizer
7	8	44960	Geneva, Switzerland	03-03-2020	@DrTedros "We can't stop #COVID19 without prot...	Neutral	geneva switzerland can stop without protect pr...
8	9	44961		04-03-2020	HI TWITTER! I am a pharmacist. I sell hand san...	Extremely Negative	twitter pharmacist sell hand sanitizer for liv...
9	10	44962	Dublin, Ireland	04-03-2020	Anyone been in a supermarket over the last few...	Extremely Positive	dublin ireland anyone be supermarket over the ...

5. GRAPHICALLY SUMMARIZE THE SENTIMENT.

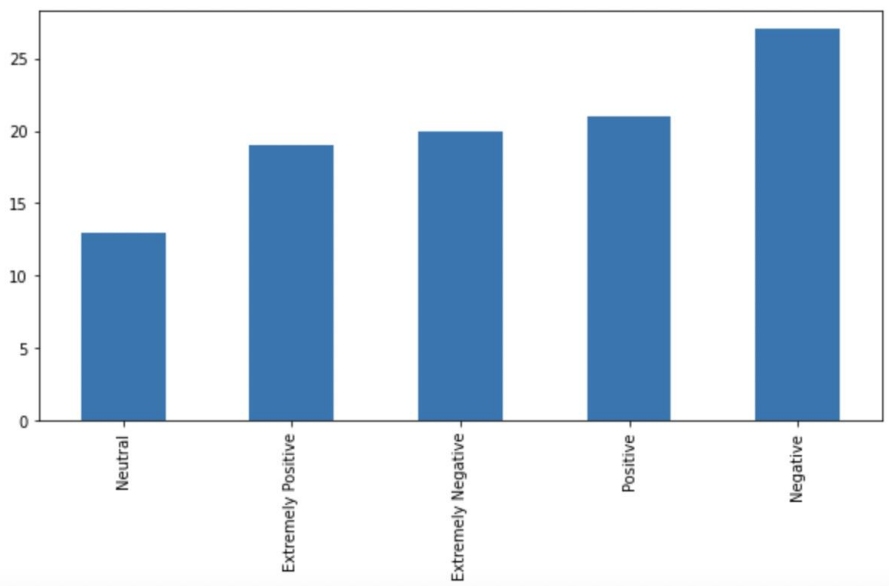
```
In [38]: Train_Data['Sentiment'].value_counts(normalize= True)

Out[38]: Negative              0.266266
         Positive              0.257257
         Neutral               0.165165
         Extremely Negative    0.163163
         Extremely Positive    0.148148
         Name: Sentiment, dtype: float64
```

```
In [39]: x = Train_Data.Sentiment.value_counts().sort_values().plot(kind = 'bar',
                                                                    figsize = (10, 5))
```




```
In [40]: Test_Data['Sentiment'].value_counts(normalize= True)

Out[40]: Negative              0.27
         Positive              0.21
         Extremely Negative    0.20
         Extremely Positive    0.19
         Neutral               0.13
         Name: Sentiment, dtype: float64
```

```
In [41]: x = Test_Data.Sentiment.value_counts().sort_values().plot(kind = 'bar',
                                                                   figsize = (10, 5))
```



6. GRAPHICALLY SUMMARIZE THE LENGTH OF THE TEXT OF THE TWEET_TEXTS USING:

- BOXPLOT
- HISTOGRAM
- DENSITY PLOT

In [42]:
```
# Before doing this problem, it is necessary to get the origional Tweet_texts data
Original_Train_Data = pd.read_csv('Corona_NLP_train.csv', encoding = 'latin-1').head(1000)
Original_Test_Data = pd.read_csv('Corona_NLP_test.csv').head(100)

Original_Train_Data = Original_Train_Data.fillna('')
Original_Test_Data = Original_Test_Data.fillna('')

Concatenation = ["Location", "TweetAt", "OriginalTweet"]
Original_Train_Data["Tweet_texts"] = Original_Train_Data[Concatenation].apply \
    (lambda row: ' '.join(row.values.astype(str)), axis=1)
Original_Test_Data["Tweet_texts"] = Original_Test_Data[Concatenation].apply \
    (lambda row: ' '.join(row.values.astype(str)), axis=1)
```

In [43]:
```
Train_Data["Len_Text_BC"] = \
    Original_Train_Data["Tweet_texts"].str.len()
Train_Data["Len_Text_AC"] = \
    Train_Data["Tweet_texts"].str.len()

pd.options.display.float_format = "{:.2f}".format
Train_Data[["Len_Text_BC", "Len_Text_AC"]].describe()
```

Out[43]:

	Len_Text_BC	Len_Text_AC
count	999.00	999.00
mean	235.38	152.62
std	65.89	57.86
min	41.00	13.00
25%	191.00	110.50
50%	246.00	159.00
75%	288.00	201.00
max	411.00	314.00

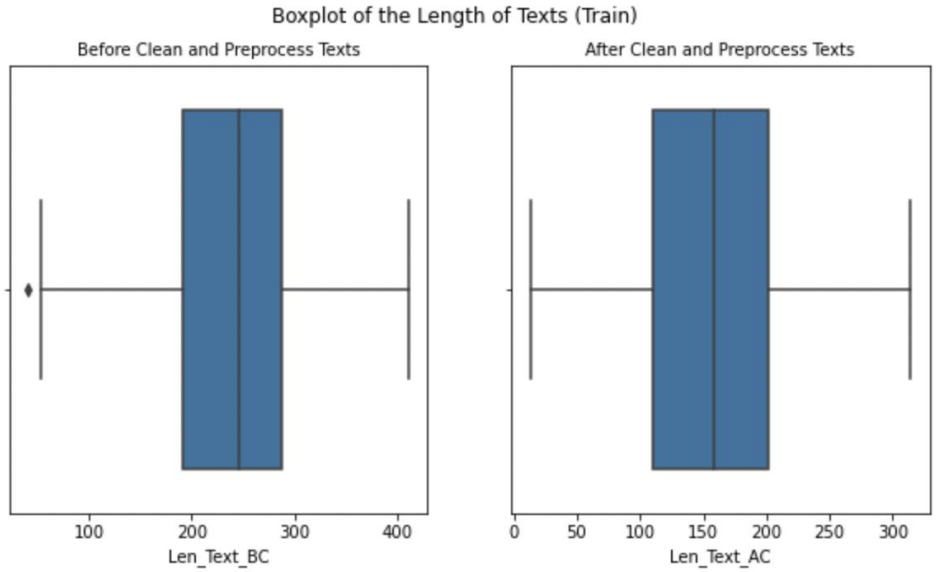
In [47]:
```
Test_Data["Len_Text_BC"] = \
    Original_Test_Data["Tweet_texts"].str.len()
Test_Data["Len_Text_AC"] = \
    Test_Data["Tweet_texts"].str.len()

pd.options.display.float_format = "{:.2f}".format
Test_Data[["Len_Text_BC", "Len_Text_AC"]].describe()
```

Out[47]:

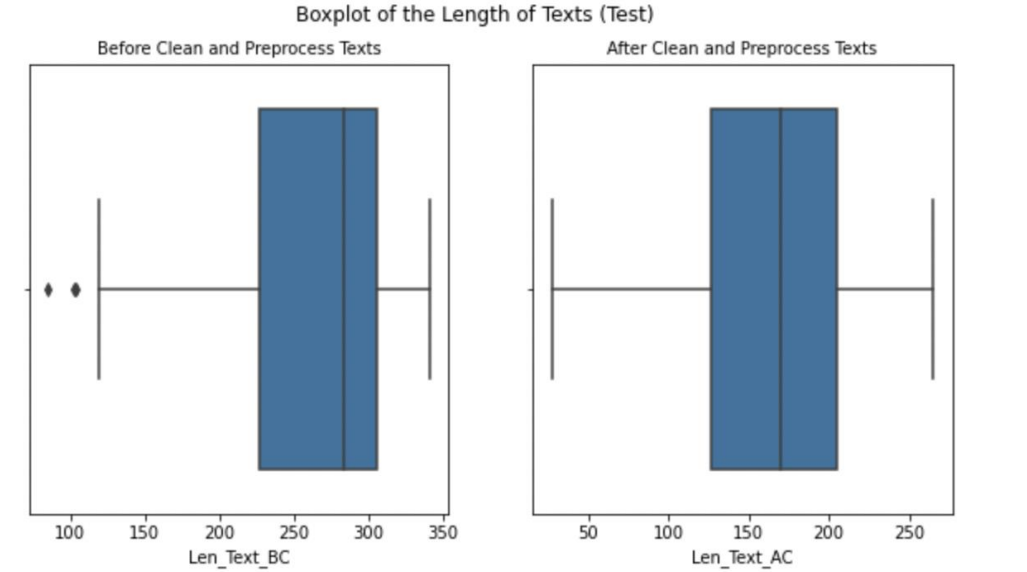
	Len_Text_BC	Len_Text_AC
count	100.00	100.00
mean	261.40	160.70
std	62.48	55.44
min	85.00	27.00
25%	226.75	126.25
50%	284.00	169.50
75%	305.50	205.25
max	341.00	265.00

6. GRAPHICALLY SUMMARIZE THE LENGTH OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [44]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Boxplot of the Length of Texts (Train)', fontsize = 12)

sns.boxplot(ax = axes[0], x = Train_Data["Len_Text_BC"])
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.boxplot(ax = axes[1], x = Train_Data["Len_Text_AC"])
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

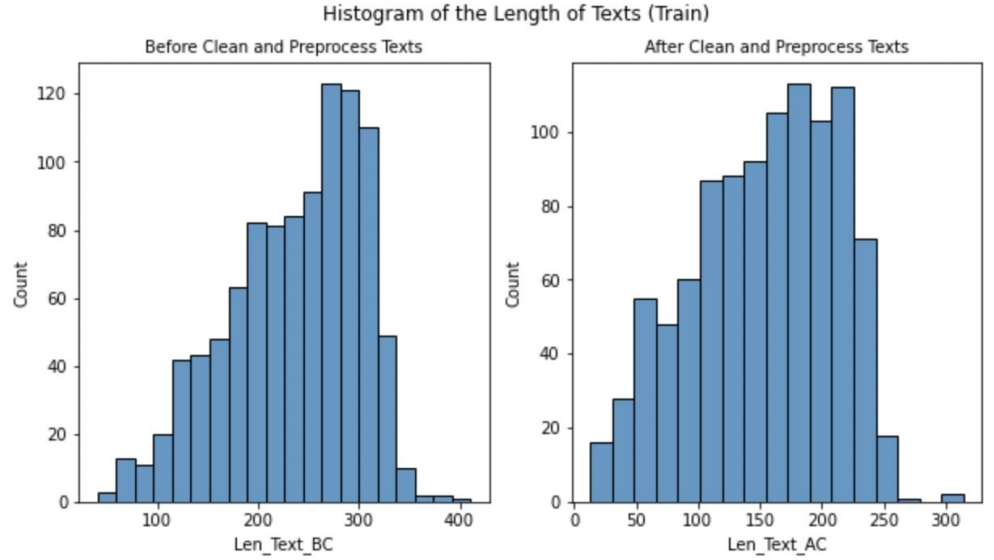


In [48]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Boxplot of the Length of Texts (Test)', fontsize = 12)

sns.boxplot(ax = axes[0], x = Test_Data["Len_Text_BC"])
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.boxplot(ax = axes[1], x = Test_Data["Len_Text_AC"])
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

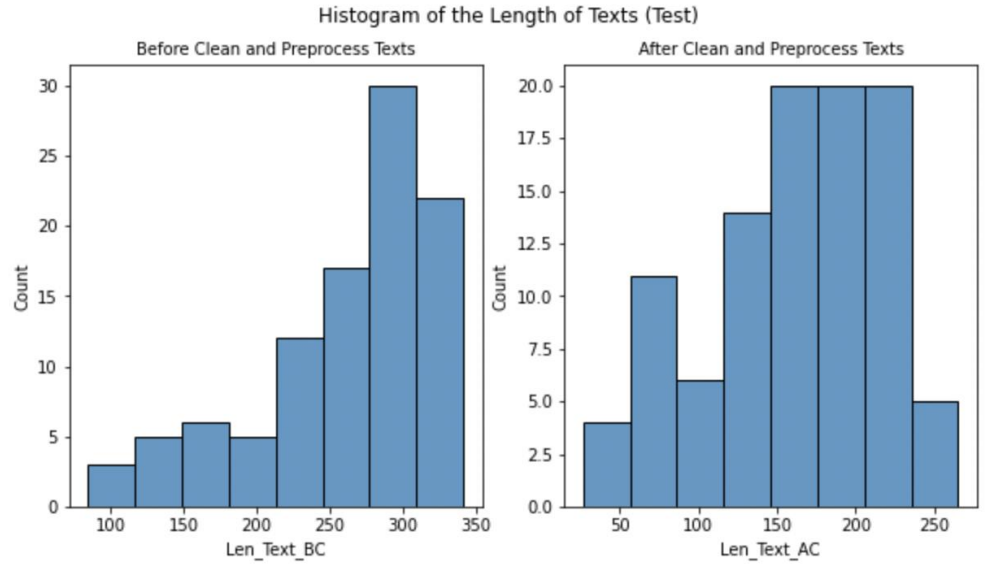


6. GRAPHICALLY SUMMARIZE THE LENGTH OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [45]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Histogram of the Length of Texts (Train)', fontsize = 12)

sns.histplot(ax = axes[0], data = Train_Data, x = "Len_Text_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.histplot(ax = axes[1], data = Train_Data, x = "Len_Text_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

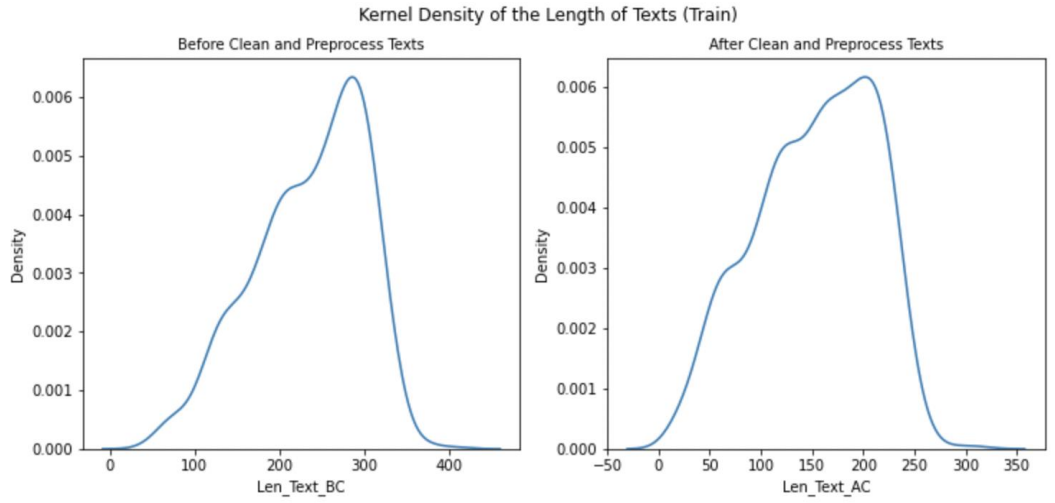


In [49]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Histogram of the Length of Texts (Test)', fontsize = 12)

sns.histplot(ax = axes[0], data = Test_Data, x = "Len_Text_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.histplot(ax = axes[1], data = Test_Data, x = "Len_Text_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

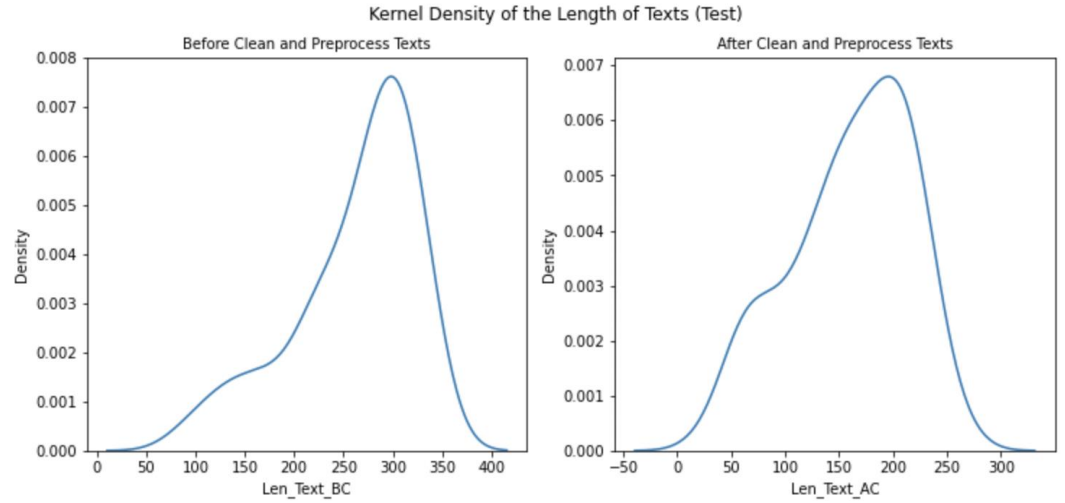


6. GRAPHICALLY SUMMARIZE THE LENGTH OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [46]:
```
fig, axes = plt.subplots(1, 2, figsize = (12, 5))
fig.suptitle('Kernel Density of the Length of Texts (Train)', fontsize = 12)

sns.kdeplot(ax = axes[0], data = Train_Data, x = "Len_Text_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.kdeplot(ax = axes[1], data = Train_Data, x = "Len_Text_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```



In [50]:
```
fig, axes = plt.subplots(1, 2, figsize = (12, 5))
fig.suptitle('Kernel Density of the Length of Texts (Test)', fontsize = 12)

sns.kdeplot(ax = axes[0], data = Test_Data, x = "Len_Text_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.kdeplot(ax = axes[1], data = Test_Data, x = "Len_Text_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```



7. GRAPHICALLY SUMMARIZE THE WORD COUNT OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [51]:
```
Train_Data["Count_Word_BC"] = \
    Original_Train_Data["Tweet_texts"].str.split().str.len()
Train_Data["Count_Word_AC"] = \
    Train_Data["Tweet_texts"].str.split().str.len()

pd.options.display.float_format = "{:.2f}".format
Train_Data[["Count_Word_BC", "Count_Word_AC"]].describe()
```

Out[51]:

	Count_Word_BC	Count_Word_AC
count	999.00	999.00
mean	35.30	25.29
std	11.82	9.95
min	7.00	2.00
25%	26.00	18.00
50%	36.00	26.00
75%	45.00	33.00
max	64.00	49.00

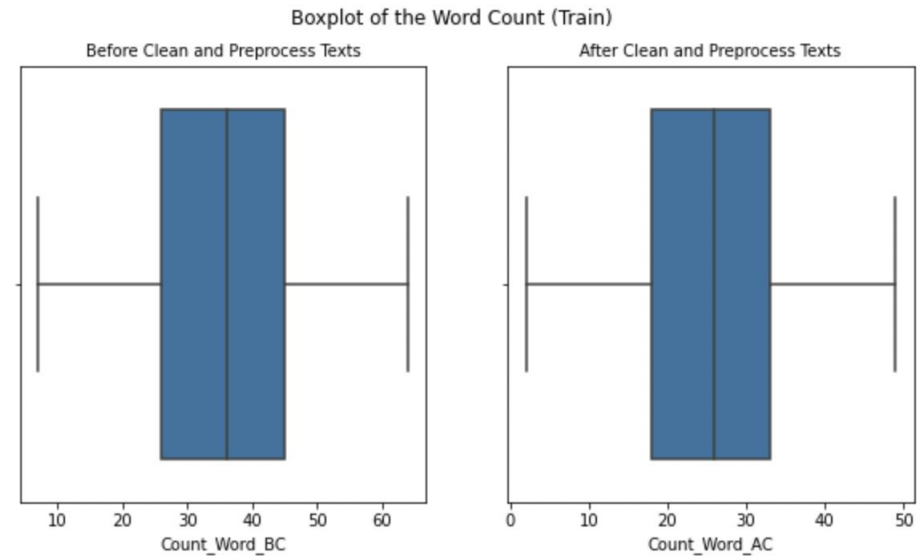
In [55]:
```
Test_Data["Count_Word_BC"] = \
    Original_Test_Data["Tweet_texts"].str.split().str.len()
Test_Data["Count_Word_AC"] = \
    Test_Data["Tweet_texts"].str.split().str.len()

pd.options.display.float_format = "{:.2f}".format
Test_Data[["Count_Word_BC", "Count_Word_AC"]].describe()
```

Out[55]:

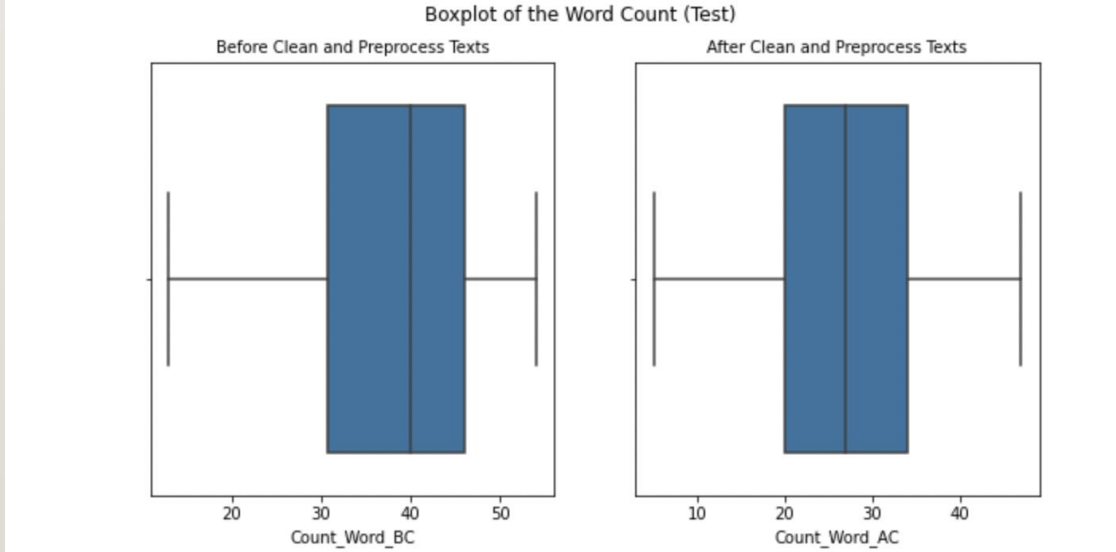
	Count_Word_BC	Count_Word_AC
count	100.00	100.00
mean	37.68	26.79
std	10.98	9.60
min	13.00	5.00
25%	30.75	20.00
50%	40.00	27.00
75%	46.00	34.00
max	54.00	47.00

7. GRAPHICALLY SUMMARIZE THE WORD COUNT OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [52]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Boxplot of the Word Count (Train)', fontsize = 12)

sns.boxplot(ax = axes[0], x = Train_Data["Count_Word_BC"])
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.boxplot(ax = axes[1], x = Train_Data["Count_Word_AC"])
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

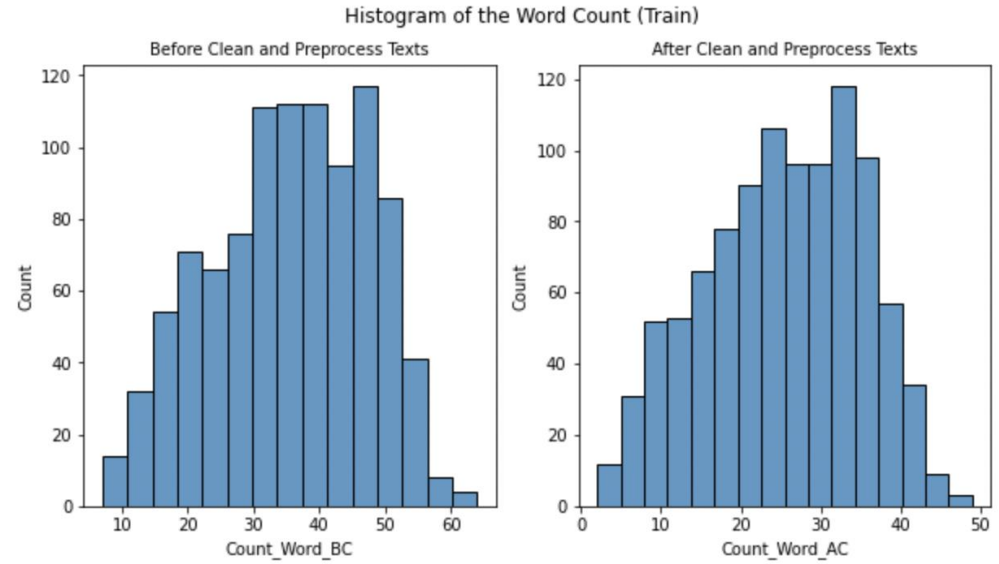


In [56]:
```
fig, axes = plt.subplots(1, 2, figsize = (10, 5))
fig.suptitle('Boxplot of the Word Count (Test)', fontsize = 12)

sns.boxplot(ax = axes[0], x = Test_Data["Count_Word_BC"])
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.boxplot(ax = axes[1], x = Test_Data["Count_Word_AC"])
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

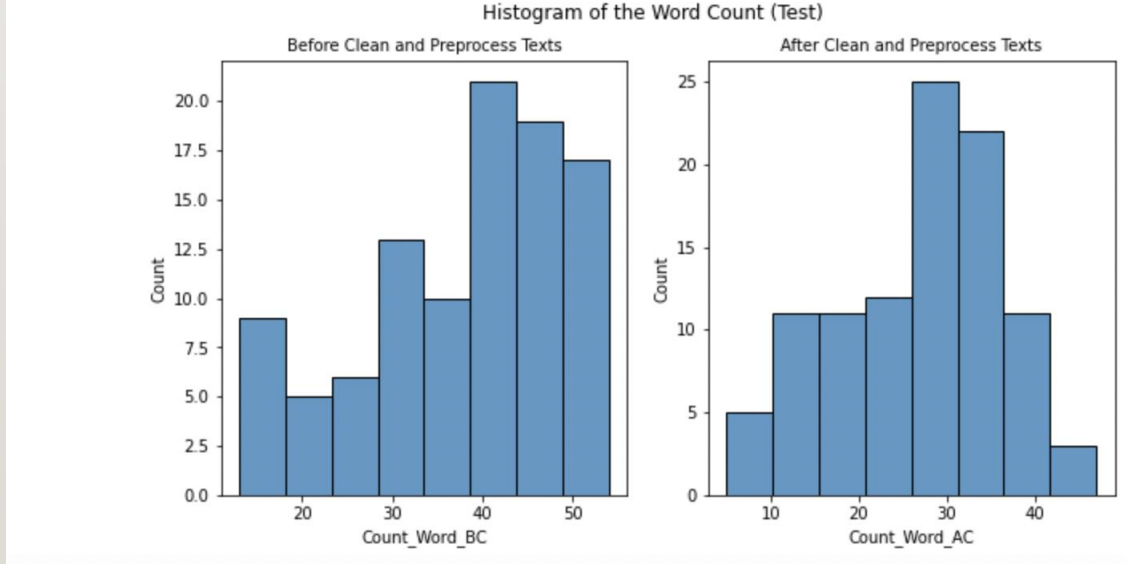


7. GRAPHICALLY SUMMARIZE THE WORD COUNT OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

```
In [53]: fig, axes = plt.subplots(1, 2, figsize = (10, 5))
         fig.suptitle('Histogram of the Word Count (Train)', fontsize = 12)

         sns.histplot(ax = axes[0], data = Train_Data, x = "Count_Word_BC")
         axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

         sns.histplot(ax = axes[1], data = Train_Data, x = "Count_Word_AC")
         x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

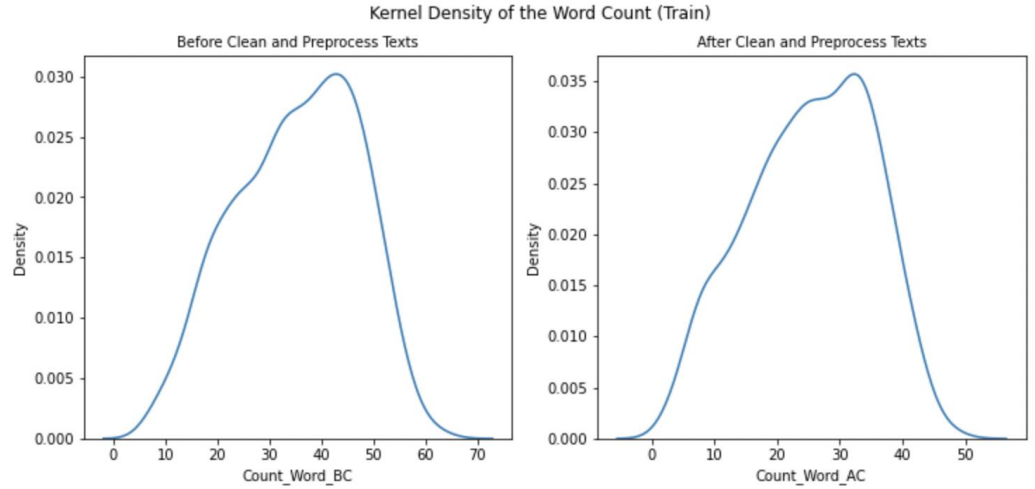



```
In [57]: fig, axes = plt.subplots(1, 2, figsize = (10, 5))
         fig.suptitle('Histogram of the Word Count (Test)', fontsize = 12)

         sns.histplot(ax = axes[0], data = Test_Data, x = "Count_Word_BC")
         axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

         sns.histplot(ax = axes[1], data = Test_Data, x = "Count_Word_AC")
         x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```

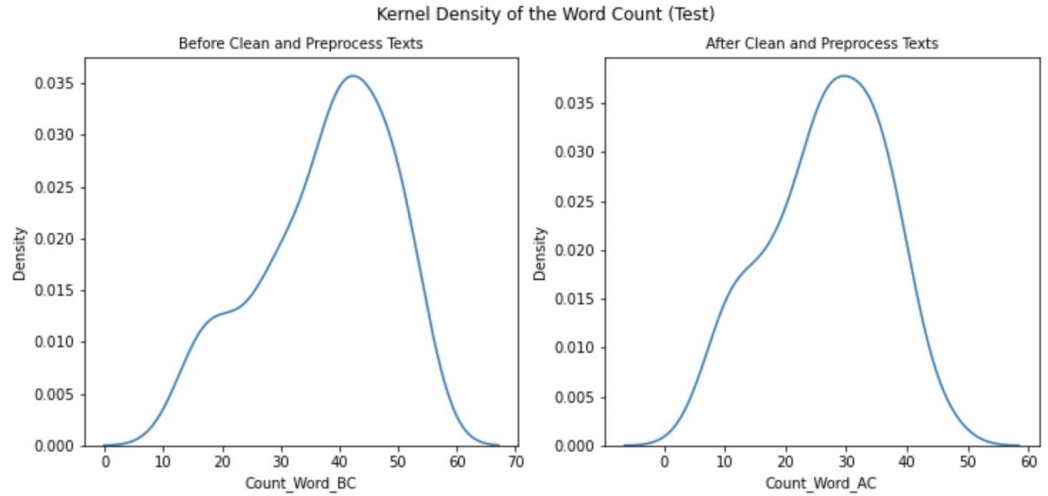


7. GRAPHICALLY SUMMARIZE THE WORD COUNT OF THE TEXT OF THE TWEET_TEXTS USING:
- BOXPLOT - HISTOGRAM - DENSITY PLOT

In [54]:
```
fig, axes = plt.subplots(1, 2, figsize = (12, 5))
fig.suptitle('Kernel Density of the Word Count (Train)', fontsize = 12)

sns.kdeplot(ax = axes[0], data = Train_Data, x = "Count_Word_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.kdeplot(ax = axes[1], data = Train_Data, x = "Count_Word_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```



In [58]:
```
fig, axes = plt.subplots(1, 2, figsize = (12, 5))
fig.suptitle('Kernel Density of the Word Count (Test)', fontsize = 12)

sns.kdeplot(ax = axes[0], data = Test_Data, x = "Count_Word_BC")
axes[0].set_title("Before Clean and Preprocess Texts", fontsize = 10)

sns.kdeplot(ax = axes[1], data = Test_Data, x = "Count_Word_AC")
x = axes[1].set_title("After Clean and Preprocess Texts", fontsize = 10)
```



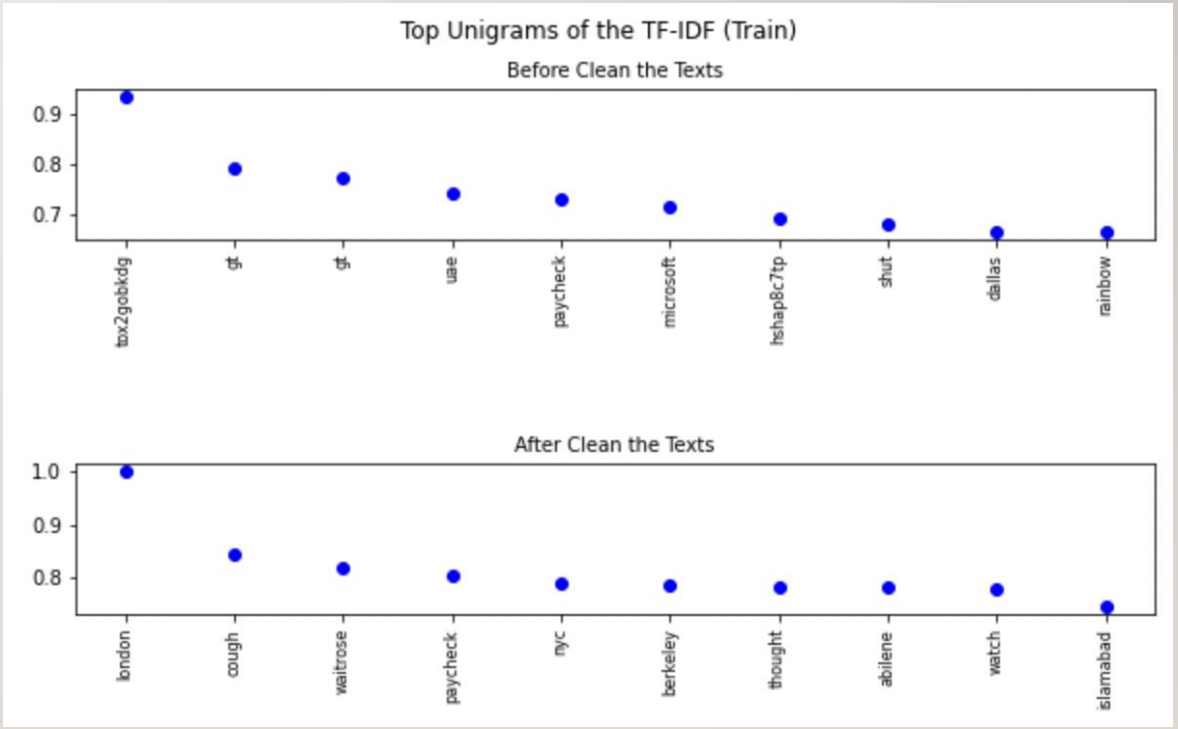
8. GRAPHICALLY SUMMARIZE THE TOP 10 UNIGRAMS AND BIGRAMS OF THE TF-IDF OF THE TEXT OF THE TWEET_TEXTS.

In [59]:
```
def get_top_tf_idf_words(df = Original_Train_Data, col = "Tweet_texts",
                         use_idf = True, ngram_range = (1, 1), top_n = 10):
    tf_idf = TfidfVectorizer(stop_words = 'english', ngram_range = ngram_range,
                                                     use_idf = use_idf)
    X_sparse_matrix = tf_idf.fit_transform(df[col])
    feature_names = np.array(tf_idf.get_feature_names())
    tf_idf_sparse_matrix = tf_idf.transform(df[col])
    sorted_idx = np.argsort(tf_idf_sparse_matrix.data)[:-(top_n+1):-1]

    return pd.DataFrame(
    {'feature': feature_names[tf_idf_sparse_matrix.indices[sorted_idx]],
     'tf_idf': tf_idf_sparse_matrix.data[sorted_idx]})
```

8. GRAPHICALLY SUMMARIZE THE TOP 10 UNIGRAMS AND BIGRAMS OF THE TF-IDF OF THE TEXT OF THE TWEET_TEXTS.

In [60]:
```
ngram_range = (1, 1)
top_n = 10

Df_Text_BC = get_top_tf_idf_words(df = Original_Train_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
Df_Text_AC = get_top_tf_idf_words(df = Train_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
```

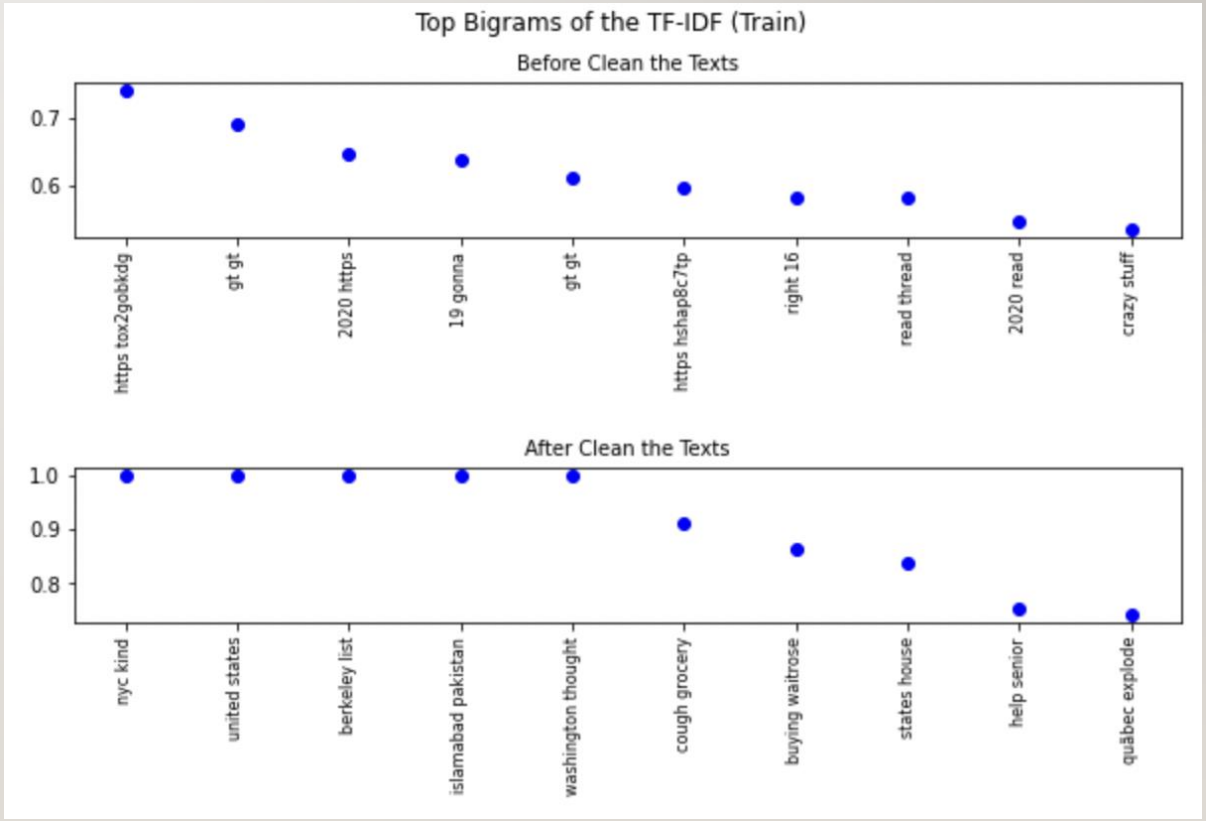
In [61]:
```
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))
fig.suptitle('Top Unigrams of the TF-IDF (Train)', fontsize = 12)

ax[0].plot(x, Df_Text_BC.tf_idf, 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Df_Text_BC.feature
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Df_Text_AC.tf_idf, 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Df_Text_AC.feature
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



8. GRAPHICALLY SUMMARIZE THE TOP 10 UNIGRAMS AND BIGRAMS OF THE TF-IDF OF THE TEXT OF THE TWEET_TEXTS.

In [62]:
```
ngram_range = (2, 2)
top_n = 10

Df_Text_BC = get_top_tf_idf_words(df = Original_Train_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
Df_Text_AC = get_top_tf_idf_words(df = Train_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
```

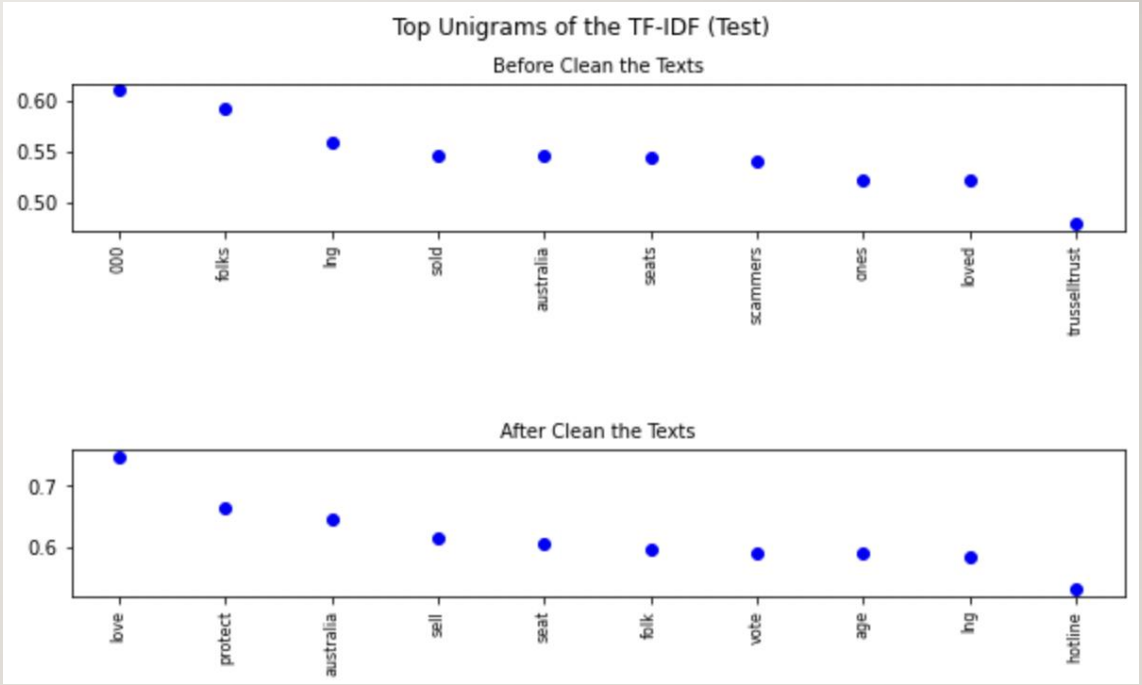
In [63]:
```
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))
fig.suptitle('Top Bigrams of the TF-IDF (Train)', fontsize = 12)

ax[0].plot(x, Df_Text_BC.tf_idf, 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Df_Text_BC.feature
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Df_Text_AC.tf_idf, 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Df_Text_AC.feature
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



8. GRAPHICALLY SUMMARIZE THE TOP 10 UNIGRAMS AND BIGRAMS OF THE TF-IDF OF THE TEXT OF THE TWEET_TEXTS.

In [64]:
```
ngram_range = (1, 1)
top_n = 10

Df_Text_BC = get_top_tf_idf_words(df = Original_Test_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
Df_Text_AC = get_top_tf_idf_words(df = Test_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
```

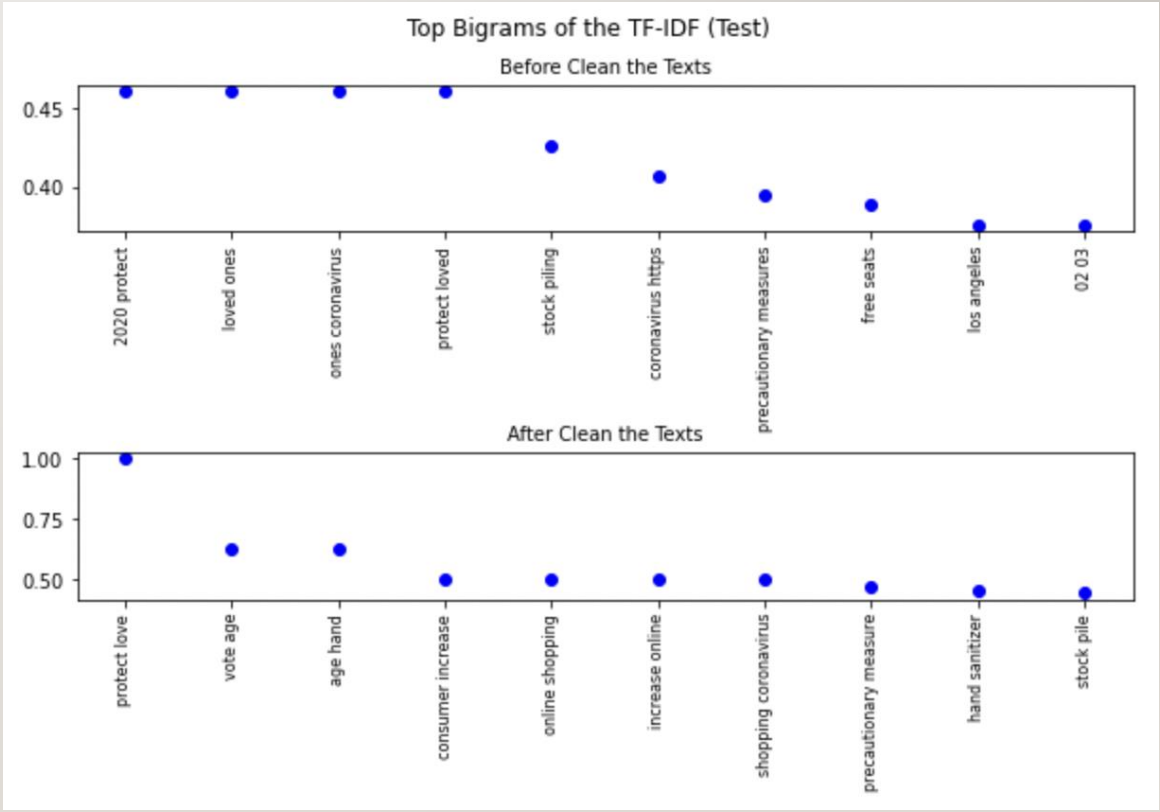
In [65]:
```
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))
fig.suptitle('Top Unigrams of the TF-IDF (Test)', fontsize = 12)

ax[0].plot(x, Df_Text_BC.tf_idf, 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Df_Text_BC.feature
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Df_Text_AC.tf_idf, 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Df_Text_AC.feature
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



8. GRAPHICALLY SUMMARIZE THE TOP 10 UNIGRAMS AND BIGRAMS OF THE TF-IDF OF THE TEXT OF THE TWEET_TEXTS.

In [66]:
```
ngram_range = (2, 2)
top_n = 10

Df_Text_BC = get_top_tf_idf_words(df = Original_Test_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
Df_Text_AC = get_top_tf_idf_words(df = Test_Data,
    col = "Tweet_texts", top_n = top_n, ngram_range = ngram_range)
```

In [67]:
```
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))
fig.suptitle('Top Bigrams of the TF-IDF (Test)', fontsize = 12)

ax[0].plot(x, Df_Text_BC.tf_idf, 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Df_Text_BC.feature
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Df_Text_AC.tf_idf, 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Df_Text_AC.feature
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

In [68]:
```
# I found that a lot of space could be saved if #9 and #10
# were done together. It was necessary to create new
# dataframes so that tweets with "Positive" and "Extremely
# Positive" sentiments could be grouped together without
# affecting the ones I currently have
```

In [69]:
```
Original_Train_Data_Positive = Original_Train_Data.copy()
Original_Train_Data_Positive.loc[Original_Train_Data_Positive["Sentiment"] \
                        == "Extremely Positive", "Sentiment"] = "Positive"

Train_Data_Positive = Train_Data.copy()
Train_Data_Positive.loc[Original_Train_Data_Positive["Sentiment"] \
                        == "Extremely Positive", "Sentiment"] = "Positive"

Original_Test_Data_Positive = Original_Test_Data.copy()
Original_Test_Data_Positive.loc[Original_Test_Data_Positive["Sentiment"] \
                        == "Extremely Positive", "Sentiment"] = "Positive"

Test_Data_Positive = Test_Data.copy()
Test_Data_Positive.loc[Original_Test_Data_Positive["Sentiment"] \
                       == "Extremely Positive", "Sentiment"] = "Positive"
```

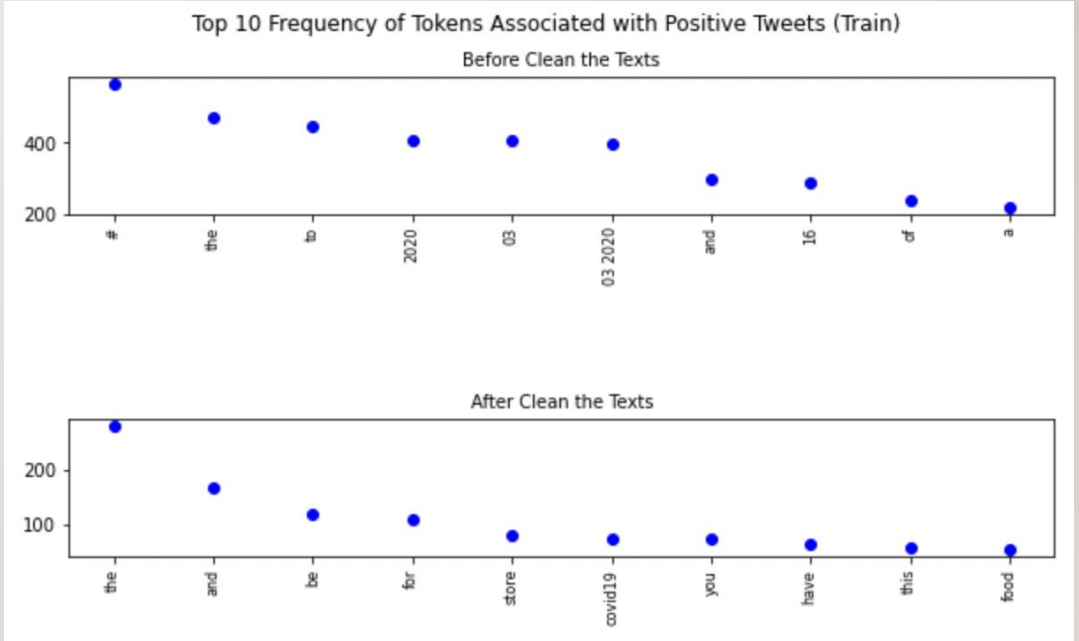
9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

In [70]:
```
Corpus_BC = st.CorpusFromPandas(Original_Train_Data_Positive,
    category_col = 'Sentiment', text_col = 'Tweet_texts', nlp = nlp).build()
Corpus_AC = st.CorpusFromPandas(Train_Data_Positive,
    category_col = 'Sentiment', text_col = 'Tweet_texts', nlp = nlp).build()
```

In [71]:
```
Term_Freq_Df_BC = Corpus_BC.get_term_freq_df()
Term_Freq_Df_BC['Positive Score'] = Corpus_BC.get_scaled_f_scores("Positive")

Term_Freq_Df_BC_Freq = Term_Freq_Df_BC.sort_values(by = 'Positive freq', ascending = False)
Term_Freq_Df_BC_Score = Term_Freq_Df_BC.sort_values(by = 'Positive Score', ascending = False)

Term_Freq_Df_AC = Corpus_AC.get_term_freq_df()
Term_Freq_Df_AC['Positive Score'] = Corpus_AC.get_scaled_f_scores("Positive")

Term_Freq_Df_AC_Freq = Term_Freq_Df_AC.sort_values(by = 'Positive freq', ascending = False)
Term_Freq_Df_AC_Score = Term_Freq_Df_AC.sort_values(by = 'Positive Score', ascending = False)
```

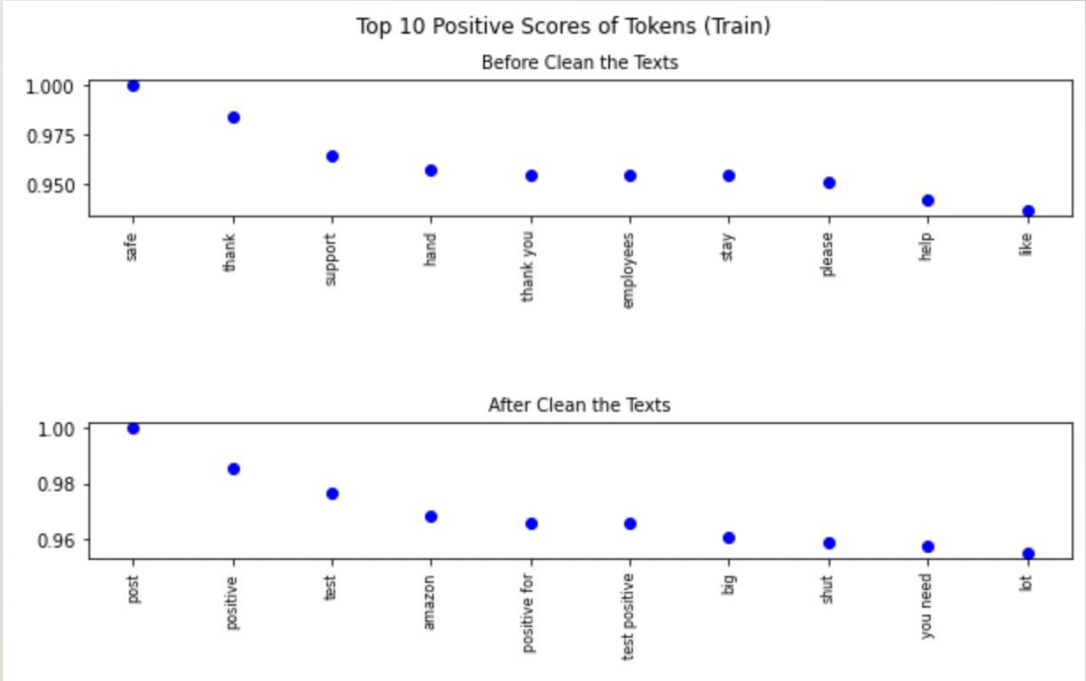
9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

In [72]:
```
top_n = 10
Term_Freq_Df_AC_Freq = Term_Freq_Df_AC_Freq.head(top_n)
Term_Freq_Df_BC_Freq = Term_Freq_Df_BC_Freq.head(top_n)
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))

fig.suptitle('Top 10 Frequency of Tokens Associated with Positive Tweets (Train)',
                                                                    fontsize= 12)

ax[0].plot(x, Term_Freq_Df_BC_Freq["Positive freq"], 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Term_Freq_Df_BC_Freq.index
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Term_Freq_Df_AC_Freq["Positive freq"], 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Term_Freq_Df_AC_Freq.index
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

```
In [73]: top_n = 10
Term_Freq_Df_AC_Score = Term_Freq_Df_AC_Score.head(top_n)
Term_Freq_Df_BC_Score = Term_Freq_Df_BC_Score.head(top_n)
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))

fig.suptitle('Top 10 Positive Scores of Tokens (Train)', fontsize = 12)

ax[0].plot(x, Term_Freq_Df_BC_Score["Positive Score"], 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Term_Freq_Df_BC_Score.index
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Term_Freq_Df_AC_Score["Positive Score"], 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Term_Freq_Df_AC_Score.index
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



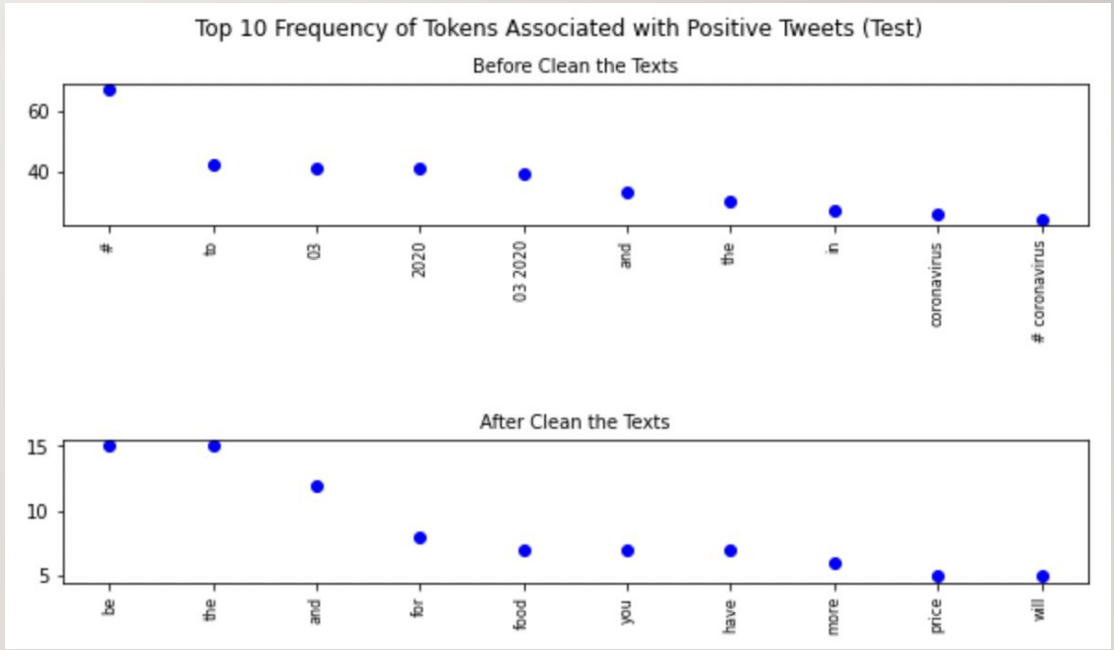
9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

In [74]:
```
Corpus_BC = st.CorpusFromPandas(Original_Test_Data_Positive,
    category_col = 'Sentiment', text_col = 'Tweet_texts', nlp = nlp).build()
Corpus_AC = st.CorpusFromPandas(Test_Data_Positive,
    category_col = 'Sentiment', text_col = 'Tweet_texts', nlp = nlp).build()
```

In [75]:
```
Term_Freq_Df_BC = Corpus_BC.get_term_freq_df()
Term_Freq_Df_BC['Positive Score'] = Corpus_BC.get_scaled_f_scores("Positive")

Term_Freq_Df_BC_Freq = Term_Freq_Df_BC.sort_values(by = 'Positive freq', ascending = False)
Term_Freq_Df_BC_Score = Term_Freq_Df_BC.sort_values(by = 'Positive Score', ascending = False)

Term_Freq_Df_AC = Corpus_AC.get_term_freq_df()
Term_Freq_Df_AC['Positive Score'] = Corpus_AC.get_scaled_f_scores("Positive")

Term_Freq_Df_AC_Freq = Term_Freq_Df_AC.sort_values(by = 'Positive freq', ascending = False)
Term_Freq_Df_AC_Score = Term_Freq_Df_AC.sort_values(by = 'Positive Score', ascending = False)
```

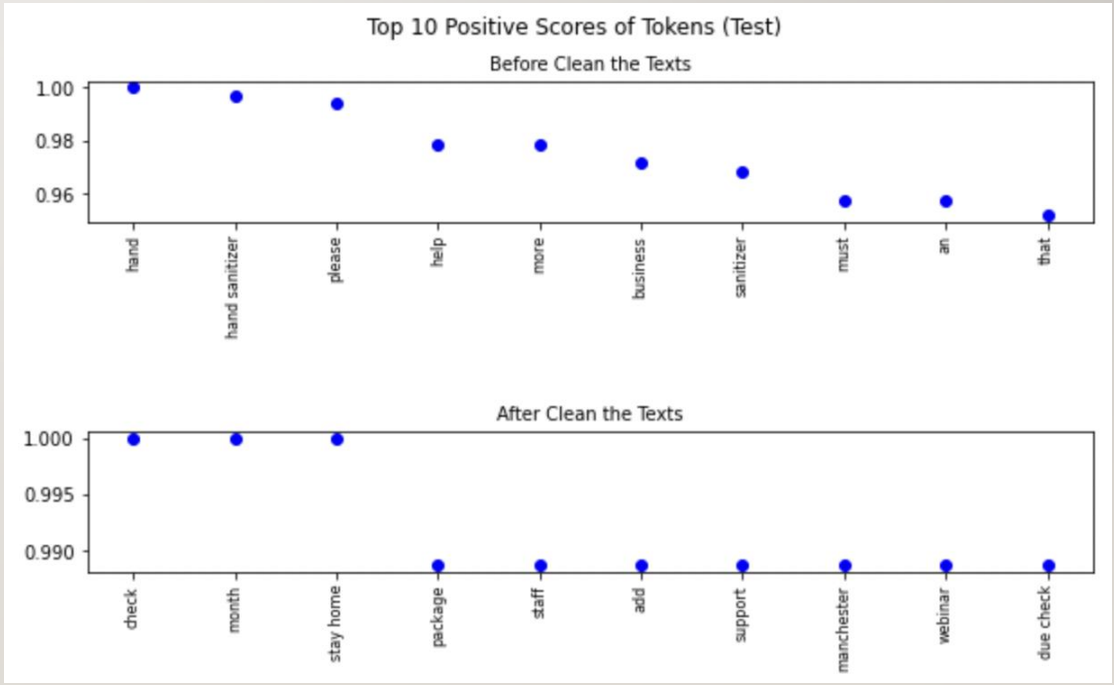
9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

In [76]:
```
top_n = 10
Term_Freq_Df_AC_Freq = Term_Freq_Df_AC_Freq.head(top_n)
Term_Freq_Df_BC_Freq = Term_Freq_Df_BC_Freq.head(top_n)
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))

fig.suptitle('Top 10 Frequency of Tokens Associated with Positive Tweets (Test)',
                                                                    fontsize= 12)

ax[0].plot(x, Term_Freq_Df_BC_Freq["Positive freq"], 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Term_Freq_Df_BC_Freq.index
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Term_Freq_Df_AC_Freq["Positive freq"], 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Term_Freq_Df_AC_Freq.index
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



9-10. VISUALIZE THE TOP 10 TERM FREQUENCY AND POSITIVE SCORES OF THE TOKENS ASSOCIATED WITH POSITIVE TWEETS USING SCATTERTEXT.

```
In [77]: top_n = 10
Term_Freq_Df_AC_Score = Term_Freq_Df_AC_Score.head(top_n)
Term_Freq_Df_BC_Score = Term_Freq_Df_BC_Score.head(top_n)
x = range(0, top_n)

fig, ax = plt.subplots(2, 1, figsize = (10, 5))

fig.suptitle('Top 10 Positive Scores of Tokens (Test)', fontsize = 12)

ax[0].plot(x, Term_Freq_Df_BC_Score["Positive Score"], 'bo')
ax[0].set_title('Before Clean the Texts', fontsize = 10)
ax[0].set_xticks(x)
x_ticks_labels = Term_Freq_Df_BC_Score.index
ax[0].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

ax[1].plot(x, Term_Freq_Df_AC_Score["Positive Score"], 'bo')
ax[1].set_title('After Clean the Texts', fontsize = 10)
ax[1].set_xticks(x)
x_ticks_labels = Term_Freq_Df_AC_Score.index
ax[1].set_xticklabels(x_ticks_labels, rotation = 'vertical', fontsize = 8)

fig.subplots_adjust(hspace = 1.5)
plt.show()
```



11. CONVERT THE COLUMN OF THE TWEET_TEXTS TO A MATRIX OF TOKEN COUNTS USING COUNTVECTORIZER AND UNIGRAMS AND BIGRAMS.

In [78]:
```
vectorizer = CountVectorizer(ngram_range = (1, 2))

matrix = vectorizer.fit_transform(Train_Data['Tweet_texts'])

print(f'The size of the feature matrix for the texts is {matrix.get_shape()}')
print(f'The first row of the feature matrix:\n{matrix[0, ]}')
print(f'There are {matrix[0, ].count_nonzero()}/{matrix.get_shape()[1]} non-zeros.')
```

```
The size of the feature matrix for the texts is (999, 22291)
The first row of the feature matrix:
  (0, 11080)    1
  (0, 1189)     2
  (0, 11082)    1
  (0, 1207)     1
There are 4/22291 non-zeros.
```

In [79]:
```
vectorizer = CountVectorizer(ngram_range = (1, 2))

matrix = vectorizer.fit_transform(Test_Data['Tweet_texts'])

print(f'The size of the feature matrix for the texts is {matrix.get_shape()}')
print(f'The first row of the feature matrix:\n{matrix[0, ]}')
print(f'There are {matrix[0, ].count_nonzero()}/{matrix.get_shape()[1]} non-zeros.')
```

```
The size of the feature matrix for the texts is (100, 3458)
The first row of the feature matrix:
  (0, 2020)     1
  (0, 3090)     1
  (0, 1960)     1
  (0, 3413)     1
  (0, 1011)     1
  (0, 1008)     1
  (0, 2759)     1
  (0, 2557)     1
  (0, 2202)     1
  (0, 3258)     1
  (0, 489)      1
```

12. PERFORM THE TF-IDF ANALYSIS ON THE COLUMN OF THE TWEET_TEXTS USING TFIDFVECTORIZER.

```
In [80]: tfidf_vectorizer = TfidfVectorizer(use_idf = True, smooth_idf = True, sublinear_tf = False)

         tf_idf_matrix = tfidf_vectorizer.fit_transform(Train_Data['Tweet_texts'])

         print(f'The size of the tf_idf matrix for the texts = {tf_idf_matrix.get_shape()}')
         print(f'The sparse tf_idf matrix is as follows:')
         print(tf_idf_matrix)
```

```
The size of the tf_idf matrix for the texts = (999, 4322)
The sparse tf_idf matrix is as follows:
  (0, 264)      0.621454220321183
  (0, 2256)     0.7834504783615812
  (1, 2703)     0.12238929509229445
  (1, 2732)     0.12527118210289453
  (1, 2619)     0.08754748155038569
  (1, 602)      0.09439848196594817
  (1, 2377)     0.2009779631404302
  (1, 3150)     0.18051910235521337
  (1, 3696)     0.11065132558117242
  (1, 161)      0.21294562950725662
  (1, 2934)     0.21294562950725662
  (1, 135)      0.17596920676740366
  (1, 3446)     0.09520718726590995
  (1, 2673)     0.09089247935272055
  (1, 3402)     0.16006024156999651
  (1, 721)      0.21294562950725662
  (1, 1298)     0.18590048664588915
```

```
In [81]: tfidf_vectorizer = TfidfVectorizer(use_idf = True, smooth_idf = True, sublinear_tf = False)

         tf_idf_matrix = tfidf_vectorizer.fit_transform(Test_Data['Tweet_texts'])

         print(f'The size of the tf_idf matrix for the texts = {tf_idf_matrix.get_shape()}')
         print(f'The sparse tf_idf matrix is as follows:')
         print(tf_idf_matrix)
```

```
The size of the tf_idf matrix for the texts = (100, 1103)
The sparse tf_idf matrix is as follows:
  (0, 913)      0.12841367475728277
  (0, 864)      0.23389872031841127
  (0, 609)      0.2548967491338323
  (0, 401)      0.2548967491338323
  (0, 449)      0.2548967491338323
  (0, 684)      0.19800235296232582
  (0, 885)      0.2548967491338323
  (0, 152)      0.2548967491338323
  (0, 1059)     0.2548967491338323
  (0, 728)      0.2548967491338323
  (0, 862)      0.23389872031841127
  (0, 926)      0.19800235296232582
  (0, 339)      0.23389872031841127
  (0, 340)      0.2548967491338323
  (0, 1098)     0.2548967491338323
  (0, 655)      0.16210598560624037
  (0, 1000)     0.2548967491338323
```

13. FIND THE COSINE SIMILARITY IN TWEET_TEXTS BETWEEN THE 200TH AND 20,000TH TWEETS.

In [82]:
```
# Because a subset of Train_Data is being used, I will instead compare the 50th and 500th tweets.
nlp = spacy.load("en_core_web_lg")

tfidf_vectorizer = TfidfVectorizer(use_idf = True, smooth_idf = True, sublinear_tf = False)

two_tweets = Train_Data.loc[[50, 500],:]
tf_idf_matrix = tfidf_vectorizer.fit_transform(two_tweets['Tweet_texts'])
print(f'The size of the tf_idf matrix for the texts = {tf_idf_matrix.get_shape()}.')
```

```
The size of the tf_idf matrix for the texts = (2, 36).
```

In [83]:
```
cos_sim = 1 - spatial.distance.cosine(tf_idf_matrix[0, :].todense(), tf_idf_matrix[1, :].todense())
print(f'The cosine similarity between "{two_tweets.loc[two_tweets.index[0], "Tweet_texts"]}" \
      and "{two_tweets.loc[two_tweets.index[1], "Tweet_texts"]}" = {cos_sim}.')
```

```
The cosine similarity between "washington everything weâ see the current covid19 outbreak have be see before previo
us epidemic and pandemic the rise fear racism panic buy food and medicine conspiracy theory the proliferation quack
cure" and "weâ still part community even when weâ ourselves stay home and stay positive" = 0.0855780603148345.
```

14. FIND THE CORPUS VECTOR EQUAL TO THE AVERAGE OF ALL THE DOCUMENT VECTORS, WHERE EACH DOCUMENT CORRESPONDS TO A TWEET OR A ROW IN THIS DATASET.

```
In [84]: corpus_vector = np.array([nlp(str(doc)).vector for doc in
                                   Train_Data['Tweet_texts']]).mean(axis=0)
         print(corpus_vector)
```

```
[-7.74068236e-02  1.18138365e-01 -9.20013934e-02 -3.83277535e-02
  1.07285246e-01 -2.91621555e-02 -3.33191641e-02 -5.98176681e-02
  3.75808217e-02  1.99290323e+00 -2.73243338e-01 -2.00557485e-02
  4.01709266e-02 -5.80984764e-02 -8.10546726e-02 -5.88076897e-02
 -8.75592828e-02  1.05457139e+00 -1.01368912e-01  3.78039479e-02
  5.48455073e-03  1.71659514e-02 -1.38569216e-03 -7.45776594e-02
 -2.91628633e-02  2.27011349e-02 -9.58650038e-02 -6.58249632e-02
  3.51534598e-02 -4.86260094e-02 -5.43451905e-02  1.04207117e-02
  6.58213953e-03  3.32086012e-02  5.70498183e-02  2.95656398e-02
  1.71042122e-02  2.17075087e-02 -2.33314801e-02 -5.06041106e-03
 -4.23441380e-02  3.24970782e-02  3.33277956e-02 -6.35451823e-02
 -1.02277100e-03  6.17574044e-02 -1.02105848e-01 -1.68599412e-02
  1.24140233e-02  2.76514795e-02 -6.45143315e-02 -2.18653642e-02
  4.45806757e-02 -8.53684731e-03  8.63378569e-02 -2.09656842e-02
 -3.13652717e-02 -7.21483752e-02 -1.21604884e-02 -7.19631165e-02
 -1.81741603e-02 -9.75581482e-02 -3.96114886e-02  1.01764038e-01
  1.26907766e-01 -7.03983232e-02 -1.82451978e-02  8.49303380e-02
  7.56720454e-02  1.20649785e-01  6.04777858e-02  3.68413478e-02
  1.58910617e-01 -1.01513667e-02  1.56328410e-01  3.81113999e-02
```

15. BUILD THE FIRST MODEL BASED ON THE TRAINING DATASET USING THE RANDOM FORESTS AND PIPELINE.

In [85]:
```
# Before building the model, it is necessary to
# define a few functions and split the data.

nlp = spacy.load("en_core_web_sm")

stop_words = spacy.lang.en.stop_words.STOP_WORDS

def spacy_tokenizer(sentence):
    mytokens = nlp(sentence)
    mytokens = [word.lemma_ for word in mytokens
                if word not in stop_words]
    return mytokens
```

In [86]:
```
# Although the text has already been cleaned, altering the code in
# this cell doesn't meaningfully change the accuracy of any of the
# models in the next few problems (except maybe the first).
# Furthermore, changing the code here could adversly affect things
# later in this notebook, so I have chosen to leave it as it is.

class features(TransformerMixin):
    def transform(self, X, **transform_params):
        return [clean_text(text) for text in X]
    def fit(self, X, y = None, **fit_params):
        return self
    def get_params(self, deep = True):
        return {}

def clean_text(text):
    text = ' '.join(re.sub("(nan)|(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])| \
                           (\w+:\/\/\S+)"," ",text).split())
    return text.strip().lower()
```

15. BUILD THE FIRST MODEL BASED ON THE TRAINING DATASET USING THE RANDOM FORESTS AND PIPELINE.

In [87]:
```
le = preprocessing.LabelEncoder()

Train_Data['Sentiment'] = le.fit_transform(Train_Data['Sentiment'])
Train_Data['Sentiment'].value_counts(normalize = True)
```

Out[87]:
```
2   0.27
4   0.26
3   0.17
0   0.16
1   0.15
Name: Sentiment, dtype: float64
```

In [88]:
```
X = Train_Data['Tweet_texts']
y = Train_Data['Sentiment']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
                                                    random_state = 42, stratify = y)
print(f'X_train dimension: {X_train.shape}; y_train dimension: {y_train.shape}')
print(f'X_test dimension: {X_test.shape}; y_train dimension: {y_test.shape}')
```

```
X_train dimension: (799,); y_train dimension: (799,)
X_test dimension: (200,); y_train dimension: (200,)
```

15. BUILD THE FIRST MODEL BASED ON THE TRAINING DATASET USING THE RANDOM FORESTS AND PIPELINE.

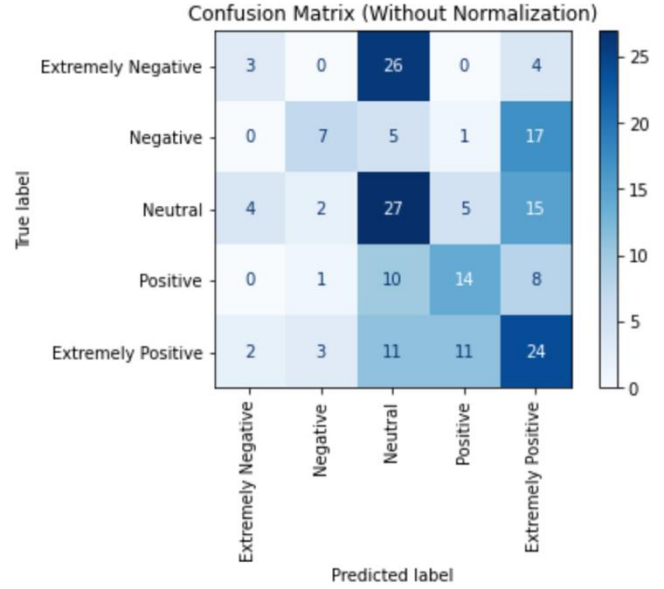
In [89]:
```
t0 = time()

tfidf_vector = TfidfVectorizer(tokenizer = spacy_tokenizer,
                               ngram_range = (1, 1))
classifier = RandomForestClassifier(random_state = 5)

pipeline = Pipeline ([("cleaner", features()), ("vectorizer",
                tfidf_vector), ("classifier", classifier)])
pipeline.fit(X_train, y_train)

print(f"It takes about {time() - t0:.1f} seconds")
```

```
It takes about 4.6 seconds
```

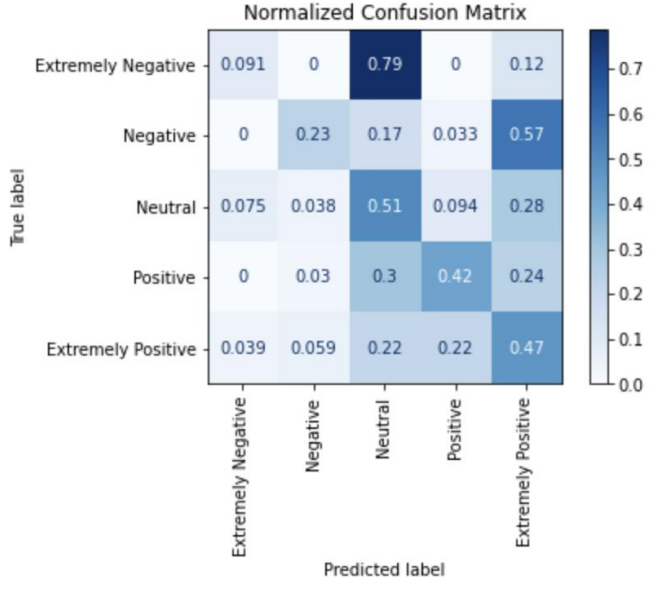
In [90]:
```
target_names = ['Extremely Negative', 'Negative', 'Neutral',
                                'Positive', "Extremely Positive"]
titles_options = [("Confusion Matrix (Without Normalization)",
            None), ("Normalized Confusion Matrix", 'true')]

for title, normalize in titles_options:
    disp = plot_confusion_matrix(pipeline, X_test, y_test,
        display_labels = target_names, cmap = plt.cm.Blues,
        normalize = normalize, xticks_rotation = 'vertical')
    disp.ax_.set_title(title)
    print(title)
    print(disp.confusion_matrix)
    plt.show()
```

15. BUILD THE FIRST MODEL BASED ON THE TRAINING DATASET USING THE RANDOM FORESTS AND PIPELINE.

```
Confusion Matrix (Without Normalization)
[[ 3  0 26  0  4]
 [ 0  7  5  1 17]
 [ 4  2 27  5 15]
 [ 0  1 10 14  8]
 [ 2  3 11 11 24]]
```




```
Normalized Confusion Matrix
[[0.09090909 0.         0.78787879 0.         0.12121212]
 [0.         0.23333333 0.16666667 0.03333333 0.56666667]
 [0.0754717  0.03773585 0.50943396 0.09433962 0.28301887]
 [0.         0.03030303 0.3030303  0.42424242 0.24242424]
 [0.03921569 0.05882353 0.21568627 0.21568627 0.47058824]]
```



15. BUILD THE FIRST MODEL BASED ON THE TRAINING DATASET USING THE RANDOM FORESTS AND PIPELINE.

In [91]:
```
y_pred = pipeline.predict(X_test)

print(classification_report(y_test, y_pred, target_names = target_names))
```

```
                    precision    recall  f1-score   support

Extremely Negative       0.33      0.09      0.14        33
          Negative       0.54      0.23      0.33        30
           Neutral       0.34      0.51      0.41        53
          Positive       0.45      0.42      0.44        33
Extremely Positive       0.35      0.47      0.40        51

          accuracy                           0.38       200
         macro avg       0.40      0.35      0.34       200
      weighted avg       0.39      0.38      0.36       200
```

16. CHECK THE FIRST MODEL ON THE TEST DATASET. IS IT A GOOD MODEL BASED ON THE SELECTED EVALUATION METRICS? PLEASE JUSTIFY YOUR ANSWER.

Judging by the precision, recall, and f-1 scores, the model is poor, as most of these values are only between 0.10 and 0.50. It should be noted that the dataset is somewhat unbalanced, so the accuracy metric could be misleading.

17. CREATE THE SECOND MODEL USING RANDOM FOREST, PIPELINE, GRID SEARCH CV FOR THE HYPERPARAMETERS FOR THE ESTIMATORS.

In [92]:
```
pipeline = Pipeline ([("cleaner", features()), ("vectorizer",
                TfidfVectorizer(tokenizer = spacy_tokenizer)),
                 ("classifier", RandomForestClassifier())])
parameters = {'vectorizer__max_df': (0.5, 1.0),
              'vectorizer__ngram_range': ((1, 1), (1, 2), (2, 2)),
              'vectorizer__use_idf': (True, False),
              'classifier__random_state': (None, 5)}
grid_search = GridSearchCV(pipeline, parameters, n_jobs = -1,
                                                 verbose = 1)

print("The pipeline contains:", [name for name,
                                          _ in pipeline.steps])
print("The parameters are as follows:")
pprint(parameters)

t0 = time()
grid_search.fit(X_train, y_train)

print(f"It takes about {time() - t0:.1f} seconds.")
print()

print(f"Best score = {grid_search.best_score_:0.3f}")
print("Best parameters set:")
best_parameters = grid_search.best_estimator_.get_params()
for param_name in sorted(parameters.keys()):
    print("\t%s: %r" % (param_name, best_parameters[param_name]))
```

```
The pipeline contains: ['cleaner', 'vectorizer', 'classifier']
The parameters are as follows:
{'classifier__random_state': (None, 5),
 'vectorizer__max_df': (0.5, 1.0),
 'vectorizer__ngram_range': ((1, 1), (1, 2), (2, 2)),
 'vectorizer__use_idf': (True, False)}
Fitting 5 folds for each of 24 candidates, totalling 120 fits
It takes about 179.3 seconds.

Best score = 0.377
Best parameters set:
        classifier__random_state: 5
        vectorizer__max_df: 0.5
        vectorizer__ngram_range: (1, 1)
        vectorizer__use_idf: False
```

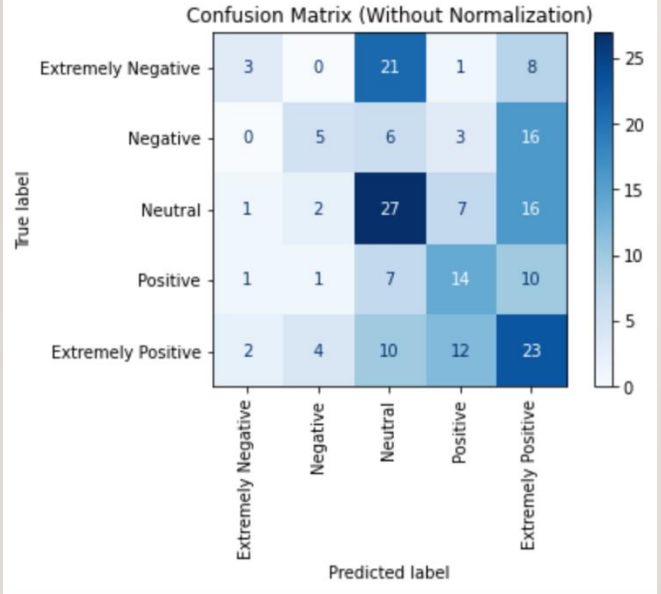
18. TUNE THE SECOND MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.

In [93]:

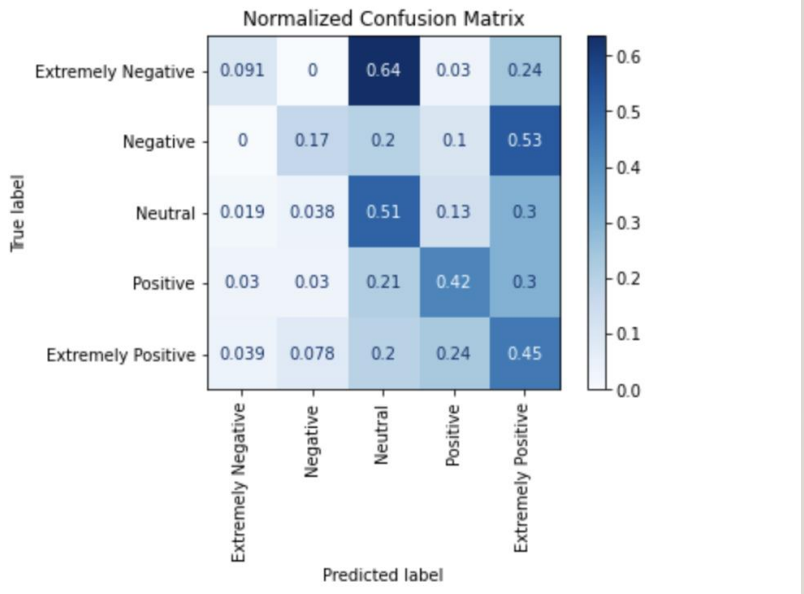
```
target_names = ['Extremely Negative', 'Negative', 'Neutral',
                                'Positive', "Extremely Positive"]
titles_options = [("Confusion Matrix (Without Normalization)",
            None), ("Normalized Confusion Matrix", 'true')]

for title, normalize in titles_options:
    disp = plot_confusion_matrix(grid_search, X_test, y_test,
        display_labels = target_names, cmap = plt.cm.Blues,
        normalize = normalize, xticks_rotation = 'vertical')
    disp.ax_.set_title(title)
    print(title)
    print(disp.confusion_matrix)
    plt.show()
```

18. TUNE THE SECOND MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.

```
Confusion Matrix (Without Normalization)
[[ 3  0 21  1  8]
 [ 0  5  6  3 16]
 [ 1  2 27  7 16]
 [ 1  1  7 14 10]
 [ 2  4 10 12 23]]
```




```
Normalized Confusion Matrix
[[0.09090909 0.         0.63636364 0.03030303 0.24242424]
 [0.         0.16666667 0.2        0.1        0.53333333]
 [0.01886792 0.03773585 0.50943396 0.13207547 0.30188679]
 [0.03030303 0.03030303 0.21212121 0.42424242 0.3030303 ]
 [0.03921569 0.07843137 0.19607843 0.23529412 0.45098039]]
```



18. TUNE THE SECOND MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.

```
In [94]: y_pred = grid_search.predict(X_test)

         print(classification_report(y_test, y_pred, target_names = target_names))
```

```
                    precision    recall  f1-score   support

Extremely Negative       0.43      0.09      0.15        33
          Negative       0.42      0.17      0.24        30
           Neutral       0.38      0.51      0.44        53
          Positive       0.38      0.42      0.40        33
Extremely Positive       0.32      0.45      0.37        51

          accuracy                           0.36       200
         macro avg       0.38      0.33      0.32       200
      weighted avg       0.38      0.36      0.34       200
```

Judging by the precision, recall, and f-1 scores, the model is poor, as most of these values are only between 0.10 and 0.50. As stated in Question 16, it should be noted that the dataset is somewhat unbalanced, so the accuracy metric could be misleading.

19. BUILD THE THIRD MODEL USING PIPELINE, GRID SEARCH CV, HYPERPARAMETER FOR THE FOLLOWING CLASSIFIERS:

- LOGISTIC REGRESSION
- SUPPORT VECTOR MACHINE
- RANDOM FOREST

In [95]:

```
class ClfSwitcher(BaseEstimator):
    def __init__(self, estimator = SGDClassifier()):
        self.estimator = estimator
    def fit(self, X, y = None, **kwargs):
        self.estimator.fit(X, y)
        return self
    def predict(self, X, y = None):
        return self.estimator.predict(X)
    def predict_proba(self, X):
        return self.estimator.predict_proba(X)
    def score(self, X, y):
        return self.estimator.score(X, y)
```

19. BUILD THE THIRD MODEL USING PIPELINE, GRID SEARCH CV, HYPERPARAMETER FOR THE FOLLOWING CLASSIFIERS:

- LOGISTIC REGRESSION
- SUPPORT VECTOR MACHINE
- RANDOM FOREST

In [96]:
```
pipeline = Pipeline ([("cleaner", features()), ("vectorizer", TfidfVectorizer()), ("classifier", ClfSwitcher())])

parameters = [{
        'vectorizer__tokenizer': [spacy_tokenizer],
        'vectorizer__max_df': [1.0],
        'vectorizer__norm': ('l1', 'l2'),
        'vectorizer__stop_words': ['english', None],

        'classifier__estimator': [LogisticRegression()],
        'classifier__estimator__solver': ['liblinear'],
        'classifier__estimator__penalty': ('l2', 'l1'),
        'classifier__estimator__max_iter': [100, 200],
        'classifier__estimator__tol': [1e-3, 1e-4]
    },{
        'vectorizer__tokenizer': [spacy_tokenizer],
        'vectorizer__max_df': [1.0],
        'vectorizer__norm': ('l1', 'l2'),
        'vectorizer__stop_words': ['english', None],

        'classifier__estimator': [svm.SVC(kernel = 'linear', C = 1.0)],
        'classifier__estimator__max_iter': [-1],
        'classifier__estimator__tol': [1e-3, 1e-4]
    },{
        'vectorizer__tokenizer': [spacy_tokenizer],
        'vectorizer__max_df': [1.0],
        'vectorizer__norm': ('l1', 'l2'),
        'vectorizer__stop_words': ['english', None],

        'classifier__estimator': [RandomForestClassifier()],
        'classifier__estimator__random_state': (None, 5)
    }]

print("The pipeline contains:", [name for name, _ in pipeline.steps])
print("parameters are as follows:")
pprint(parameters)

t0 = time()
gscv = GridSearchCV(pipeline, parameters, cv = 5, n_jobs = -1, return_train_score = False, verbose = 3)
gscv.fit(X_train, y_train)
print(f"It takes about {time() - t0:.3f} seconds")
```

```
The pipeline contains: ['cleaner', 'vectorizer', 'classifier']
parameters are as follows:
[{'classifier__estimator': [LogisticRegression()],
  'classifier__estimator__max_iter': [100, 200],
  'classifier__estimator__penalty': ('l2', 'l1'),
  'classifier__estimator__solver': ['liblinear'],
  'classifier__estimator__tol': [0.001, 0.0001],
  'vectorizer__max_df': [1.0],
  'vectorizer__norm': ('l1', 'l2'),
  'vectorizer__stop_words': ['english', None],
  'vectorizer__tokenizer': [<function spacy_tokenizer at 0x7ff4312eee50>]},
 {'classifier__estimator': [SVC(kernel='linear')],
  'classifier__estimator__max_iter': [-1],
  'classifier__estimator__tol': [0.001, 0.0001],
  'vectorizer__max_df': [1.0],
  'vectorizer__norm': ('l1', 'l2'),
  'vectorizer__stop_words': ['english', None],
  'vectorizer__tokenizer': [<function spacy_tokenizer at 0x7ff4312eee50>]},
 {'classifier__estimator': [RandomForestClassifier()],
  'classifier__estimator__random_state': (None, 5),
  'vectorizer__max_df': [1.0],
  'vectorizer__norm': ('l1', 'l2'),
  'vectorizer__stop_words': ['english', None],
  'vectorizer__tokenizer': [<function spacy_tokenizer at 0x7ff4312eee50>]}]
Fitting 5 folds for each of 48 candidates, totalling 240 fits
```

19. BUILD THE THIRD MODEL USING PIPELINE, GRID SEARCH CV, HYPERPARAMETER FOR THE FOLLOWING CLASSIFIERS:

- LOGISTIC REGRESSION
- SUPPORT VECTOR MACHINE
- RANDOM FOREST

In [97]:
```
print(f"Best score= {gscv.best_score_:0.3f}")

best_parameters = gscv.best_estimator_.get_params()

all_classifiers = []
for parameter in parameters:
    all_classifiers.append(parameter['classifier__estimator'])
all_classifiers = [str(alg) for clf in all_classifiers for alg in clf]
print("All potential classifiers:")
pprint(all_classifiers)

idx = all_classifiers.index(str(best_parameters['classifier__estimator']))
print("Best parameters set:")
for param_name in sorted(parameters[idx].keys()):
    print("\t%s: %r" % (param_name, best_parameters[param_name]))
```

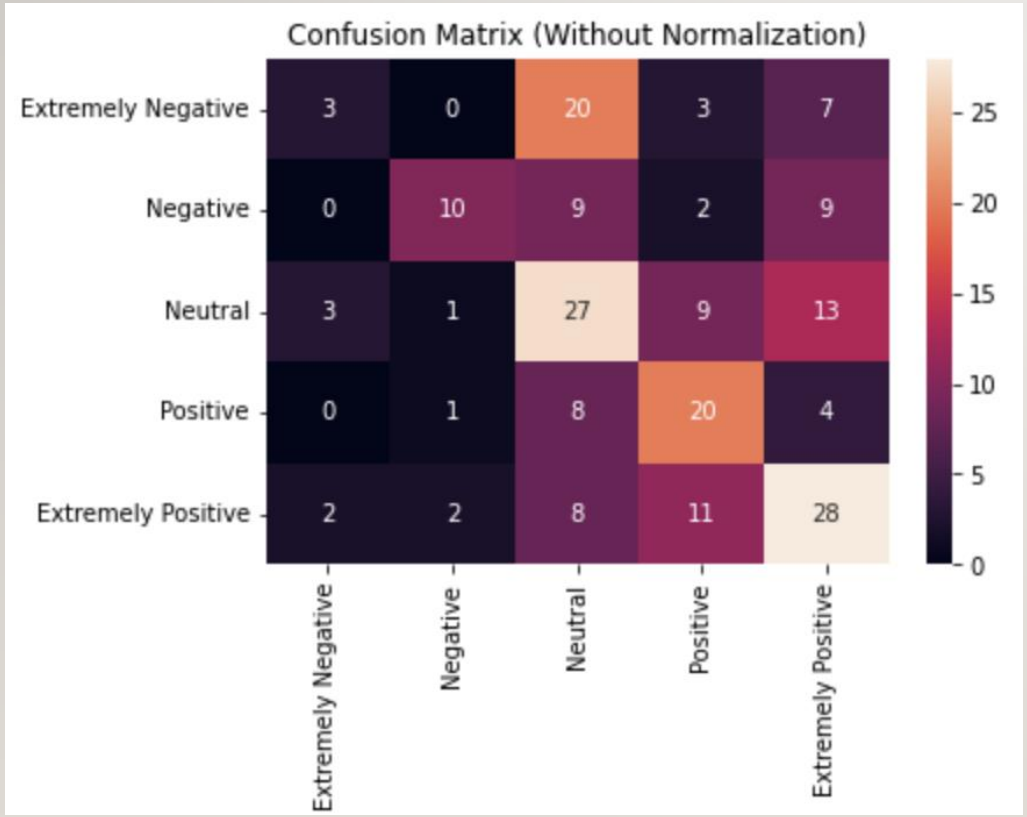
```
Best score= 0.365
All potential classifiers:
['LogisticRegression()',
 "SVC(kernel='linear')",
 'RandomForestClassifier(random_state=5)']
Best parameters set:
        classifier__estimator: RandomForestClassifier(random_state=5)
        classifier__estimator__random_state: 5
        vectorizer__max_df: 1.0
        vectorizer__norm: 'l1'
        vectorizer__stop_words: 'english'
        vectorizer__tokenizer: <function spacy_tokenizer at 0x7ff4312eee50>
```

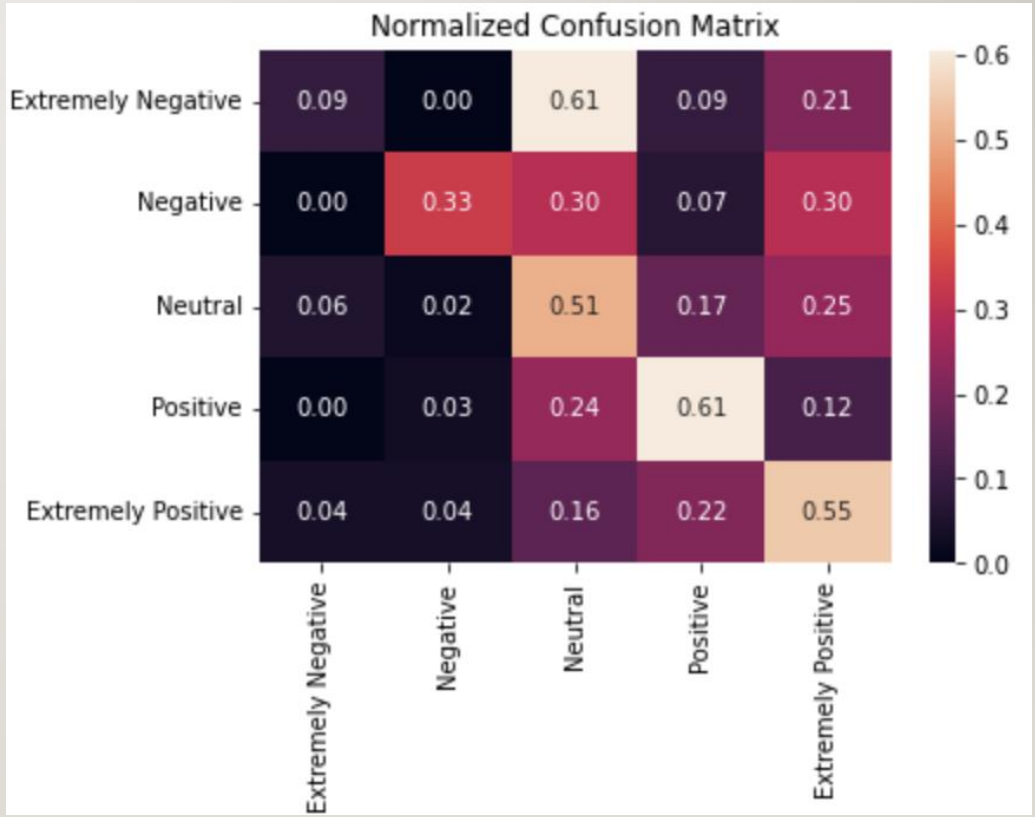
20. TUNE THE THIRD MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.

In [98]:

```
y_pred = gscv.predict(X_test)

cm = confusion_matrix(y_test, y_pred)
ax = plt.axes()
sns.heatmap(cm, annot = True, fmt = 'd',
    xticklabels = target_names, yticklabels = target_names)
ax.set_title("Confusion Matrix (Without Normalization)")
plt.show()

cm = cm / cm.astype(np.float64).sum(axis=1)[:,None]
ax = plt.axes()
sns.heatmap(cm, annot = True, fmt = '.2f',
    xticklabels = target_names, yticklabels = target_names)
ax.set_title("Normalized Confusion Matrix")
plt.show()
```

20. TUNE THE THIRD MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.





20. TUNE THE THIRD MODEL AND PERFORM MODEL DIAGNOSTICS ON THE TEST DATASET. IS IT A GOOD MODEL? PLEASE JUSTIFY YOUR ANSWER.

```
In [99]: print(classification_report(y_test, y_pred, target_names = target_names))
```

	precision	recall	f1-score	support
Extremely Negative	0.38	0.09	0.15	33
Negative	0.71	0.33	0.45	30
Neutral	0.38	0.51	0.43	53
Positive	0.44	0.61	0.51	33
Extremely Positive	0.46	0.55	0.50	51
accuracy			0.44	200
macro avg	0.47	0.42	0.41	200
weighted avg	0.46	0.44	0.42	200

Judging by the precision, recall, and f-1 scores, the model is poor, as most of these values are only between 0.30 and 0.60. As stated in Questions 16 and 18, it should be noted that the dataset is somewhat unbalanced, so the accuracy metric could be misleading.

21. GENERATE THE FIRST TOPIC MODEL BY SETTING FIVE TOPICS USING LDA METHOD AND COUNTVECTORIZER.

In [100]:
```
tf_vectorizer = CountVectorizer(max_df = 0.90, min_df = 5, stop_words = 'english')
document_word_matrix_tf = tf_vectorizer.fit_transform(Train_Data['Tweet_texts'])

lda = LatentDirichletAllocation(n_components = 5)
lda.fit(document_word_matrix_tf)
```

Out[100]:
```
LatentDirichletAllocation
LatentDirichletAllocation(n_components=5)
```

22. PLOT THE TOP 15 WORDS FOR EACH TOPIC FOR THE MODEL ABOVE.

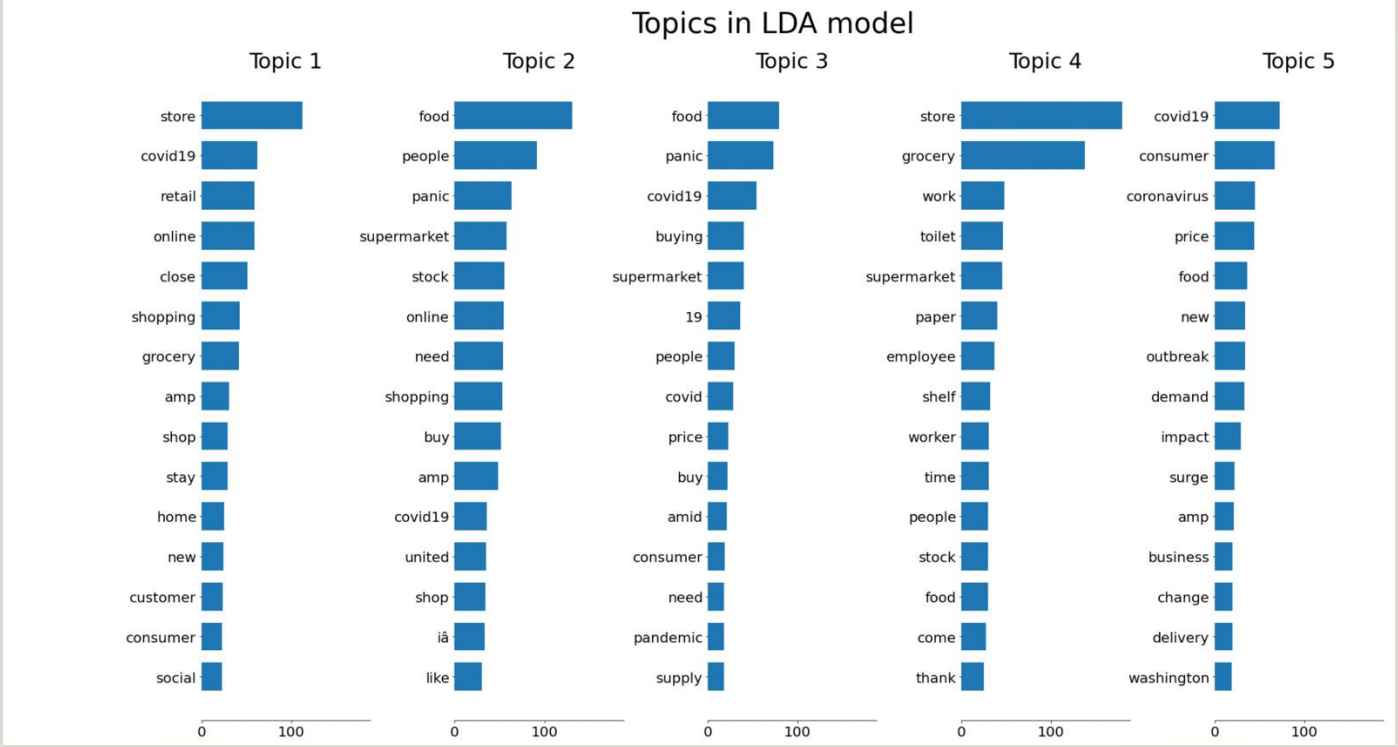
In [101]:
```
n_top_words = 15

for index, topic in enumerate(lda.components_):
    print(f'THE TOP {n_top_words} WORDS FOR TOPIC #{index + 1}')
    print([tf_vectorizer.get_feature_names()[j] for j in topic.argsort()[-n_top_words:]])
    print('')
```

```
THE TOP 15 WORDS FOR TOPIC #1
['social', 'consumer', 'customer', 'new', 'home', 'stay', 'shop', 'amp', 'grocery', 'shopping', 'close', 'online',
'retail', 'covid19', 'store']

THE TOP 15 WORDS FOR TOPIC #2
['like', 'iâ', 'shop', 'united', 'covid19', 'amp', 'buy', 'shopping', 'need', 'online', 'stock', 'supermarket', 'pa
nic', 'people', 'food']

THE TOP 15 WORDS FOR TOPIC #3
['supply', 'pandemic', 'need', 'consumer', 'amid', 'buy', 'price', 'covid', 'people', '19', 'supermarket', 'buying'
, 'covid19', 'panic', 'food']

THE TOP 15 WORDS FOR TOPIC #4
['thank', 'come', 'food', 'stock', 'people', 'time', 'worker', 'shelf', 'employee', 'paper', 'supermarket', 'toilet
', 'work', 'grocery', 'store']

THE TOP 15 WORDS FOR TOPIC #5
['washington', 'delivery', 'change', 'business', 'amp', 'surge', 'impact', 'demand', 'outbreak', 'new', 'food', 'pr
ice', 'coronavirus', 'consumer', 'covid19']
```

In [102]:
```
def plot_top_words(model, feature_names, n_top_words, title):
    fig, axes = plt.subplots(1, 5, figsize=(30, 15), sharex=True)
    axes = axes.flatten()

    for topic_idx, topic in enumerate(model.components_):
        top_features_ind = topic.argsort()[:-n_top_words - 1:-1]
        top_features = [feature_names[i] for i in top_features_ind]
        weights = topic[top_features_ind]
        ax = axes[topic_idx]
        ax.barh(top_features, weights, height = 0.7)
        ax.set_title(f'Topic {topic_idx + 1}', fontdict = {'fontsize': 30})
        ax.invert_yaxis()
        ax.tick_params(axis = 'both', which = 'major', labelsize = 20)

        for i in 'top right left'.split():
            ax.spines[i].set_visible(False)

        fig.suptitle(title, fontsize = 40)

    plt.subplots_adjust(top = 0.90, bottom = 0.05, wspace = 0.5, hspace = 0.3)
    plt.show()
```

22. PLOT THE TOP 15 WORDS FOR EACH TOPIC FOR THE MODEL ABOVE.

```
In [103]: tf_feature_names = tf_vectorizer.get_feature_names()
          plot_top_words(lda, tf_feature_names, n_top_words, 'Topics in LDA model')
```



23. GENERATE THE SECOND TOPIC MODEL BY SETTING FIVE TOPICS USING LDA METHOD AND TFIDFVECTORIZER.

In [104]:
```
tfidf_vectorizer = TfidfVectorizer(stop_words = 'english',
    lowercase = True, token_pattern = r'\b[a-zA-Z0-9]{2,}\b',
                                       max_df = 0.9, min_df = 5)
document_word_matrix_tf_idf = \
    tfidf_vectorizer.fit_transform(Train_Data['Tweet_texts'])

lda_tf_idf = LatentDirichletAllocation(n_components = 5)
lda_tf_idf.fit(document_word_matrix_tf_idf)
```

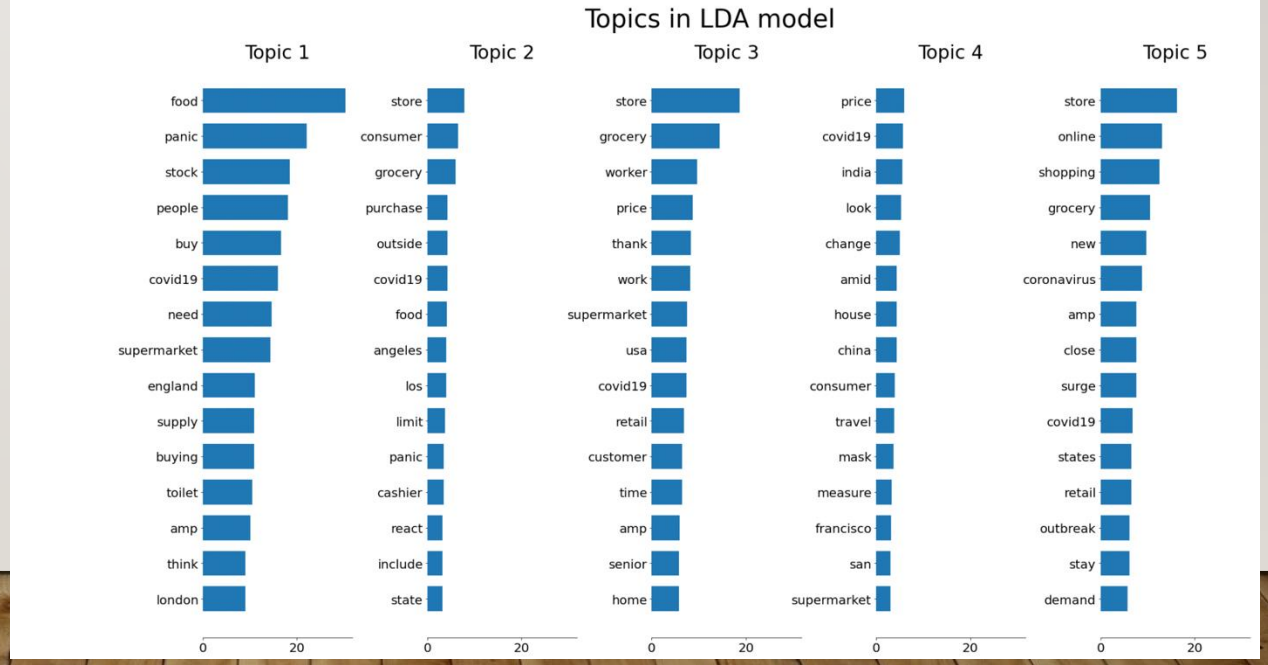
Out[104]:
```
LatentDirichletAllocation
LatentDirichletAllocation(n_components=5)
```

24. PLOT THE TOP 15 WORDS FOR EACH TOPIC FOR THE SECOND MODEL ABOVE.

```
In [105]: n_top_words = 15

for index, topic in enumerate(lda_tf_idf.components_):
    print(f'THE TOP {n_top_words} WORDS FOR TOPIC #{index + 1}')
    print([tfidf_vectorizer.get_feature_names()[j] for j in topic.argsort()[-n_top_words:]])
    print('')
```

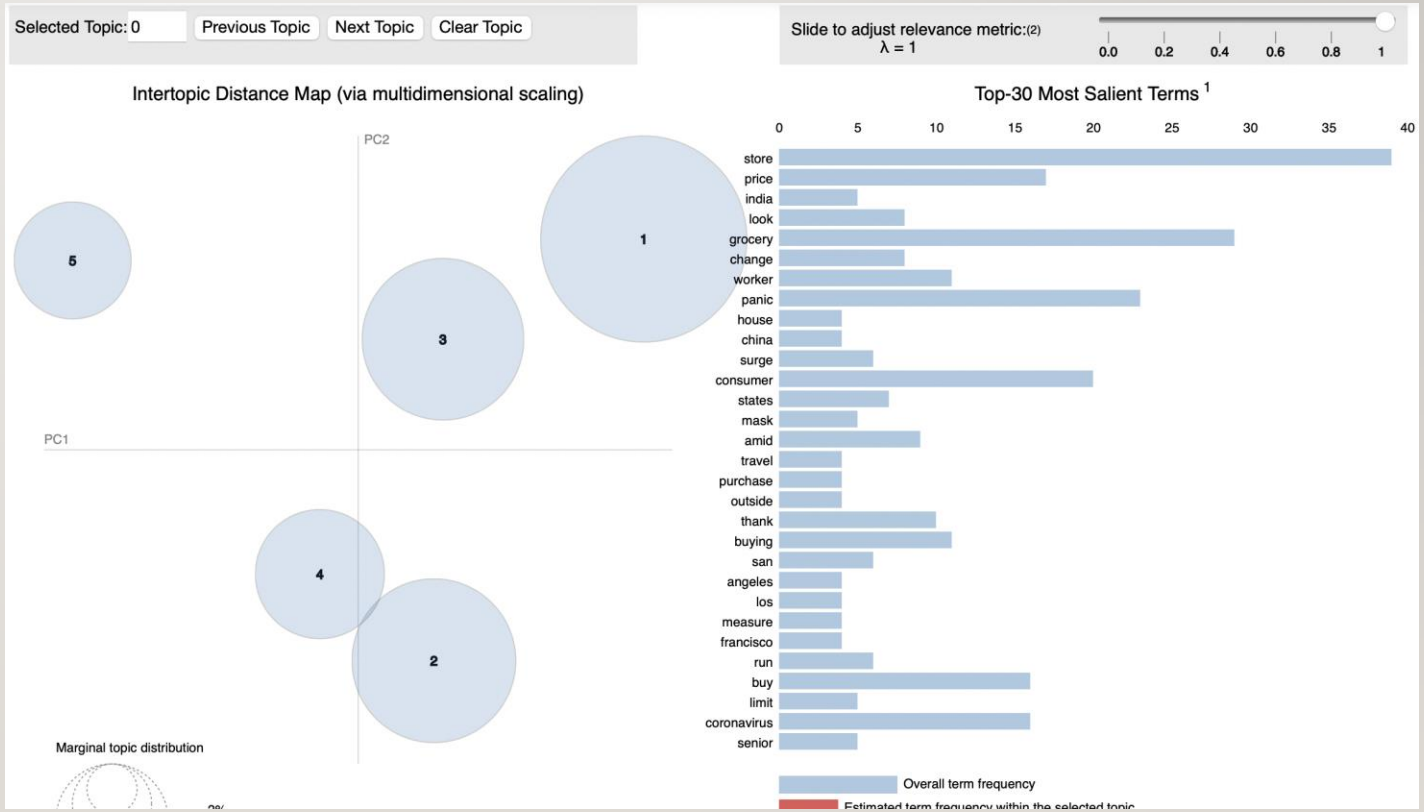
```
THE TOP 15 WORDS FOR TOPIC #1
['london', 'think', 'amp', 'toilet', 'buying', 'supply', 'england', 'supermarket', 'need', 'covid19', 'buy', 'peopl
e', 'stock', 'panic', 'food']

THE TOP 15 WORDS FOR TOPIC #2
['state', 'include', 'react', 'cashier', 'panic', 'limit', 'los', 'angeles', 'food', 'covid19', 'outside', 'purchas
e', 'grocery', 'consumer', 'store']

THE TOP 15 WORDS FOR TOPIC #3
['home', 'senior', 'amp', 'time', 'customer', 'retail', 'covid19', 'usa', 'supermarket', 'work', 'thank', 'price',
'worker', 'grocery', 'store']

THE TOP 15 WORDS FOR TOPIC #4
['supermarket', 'san', 'francisco', 'measure', 'mask', 'travel', 'consumer', 'china', 'house', 'amid', 'change', 'l
ook', 'india', 'covid19', 'price']

THE TOP 15 WORDS FOR TOPIC #5
['demand', 'stay', 'outbreak', 'retail', 'states', 'covid19', 'surge', 'close', 'amp', 'coronavirus', 'new', 'groce
ry', 'shopping', 'online', 'store']
```

```
In [106]: tf_feature_names = tfidf_vectorizer.get_feature_names()
plot_top_words(lda_tf_idf, tf_feature_names, n_top_words, 'Topics in LDA model')
```



25. VISUALIZE THE SECOND TOPIC MODEL USING THE DIMENSION REDUCTION METHOD.

```
In [107]: pyLDAvis.enable_notebook()

          pyLDAvis.sklearn.prepare(lda_model = lda_tf_idf,
              dtm = document_word_matrix_tf_idf,
                  vectorizer = tfidf_vectorizer)
```



QUESTIONS?