In [1]:

```
import pandas as pd
import numpy as np
import nltk
import wordcloud
import re
import string
from nltk.corpus import stopwords

from sklearn.feature_extraction.text import TfidfVectorizer

import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
```

In [2]:

```
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Yogesh\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\Yogesh\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[2]:

```
True
```

In [3]:

```
df = pd.read_csv('Twitter_Data.csv')
```

In [4]:

```
df.shape
```

Out[4]:

```
(162980, 2)
```

In [5]:

```
df.head()
```

Out[5]:

	clean_text	category
0	when modi promised “minimum government maximum...	-1.0
1	talk all the nonsense and continue all the dra...	0.0
2	what did just say vote for modi welcome bjp t...	1.0
3	asking his supporters prefix chowkidar their n...	1.0
4	answer who among these the most powerful world...	1.0

In [6]:

```
df.category.value_counts()
```

Out[6]:

```
 1.0    72250
 0.0    55213
-1.0    35510
Name: category, dtype: int64
```

In [7]:

```
df['category']=df['category'].map({-1.0:'Negative', 0.0:'Neutral', 1.0:'Positive'})
```

In [8]:

```
df.head()
```

Out[8]:

	clean_text	category
0	when modi promised “minimum government maximum...	Negative
1	talk all the nonsense and continue all the dra...	Neutral
2	what did just say vote for modi welcome bjp t...	Positive
3	asking his supporters prefix chowkidar their n...	Positive
4	answer who among these the most powerful world...	Positive

In [9]:

```
df.isna().sum()
```

Out[9]:

```
clean_text    4
category      7
dtype: int64
```

In [10]:

```
df = df.dropna()
```

In [11]:

```
df.isna().sum()
```

Out[11]:

```
clean_text    0
category      0
dtype: int64
```

In [12]:

```
punct = string.punctuation
punct
```

Out[12]:

```
'!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~'
```

In [13]:

```
stopWords = stopwords.words('english')
stopWords
```

```
 'yourself',
 'yourselves',
 'he',
 'him',
 'his',
 'himself',
 'she',
 "she's",
 'her',
 'hers',
 'herself',
 'it',
 "it's",
 'its',
 'itself',
 'they',
 'them',
 'their',
 'theirs',
 'themselves',
```

In [14]:

```
ps = nltk.PorterStemmer()
wn = nltk.WordNetLemmatizer()
```

In [15]:

```
def cleanData(text):


    text = text.lower()


    text = re.sub(r"[^A-Za-z0-9]",' ', text)


    text = ''.join([char for char in text if char not in punct])


    text = [wn.lemmatize(word) for word in text.split(' ') if ((word not in stopWords) & len(word)!=(

    return ' '.join(text)
```

In [16]:

```
df
```

Out[16]:

	clean_text	category
0	when modi promised “minimum government maximum...	Negative
1	talk all the nonsense and continue all the dra...	Neutral
2	what did just say vote for modi welcome bjp t...	Positive
3	asking his supporters prefix chowkidar their n...	Positive
4	answer who among these the most powerful world...	Positive
...	...	...
162975	why these 456 crores paid neerav modi not reco...	Negative
162976	dear rss terrorist payal gawar what about modi...	Negative
162977	did you cover her interaction forum where she ...	Neutral
162978	there big project came into india modi dream p...	Neutral
162979	have you ever listen about like gurukul where ...	Positive

162969 rows × 2 columns

In [17]:

```
df["clean_text"][0]
```

Out[17]:

```
'when modi promised “minimum government maximum governance” expected him begin the dif
ficult job reforming the state why does take years get justice state should and not bu
siness and should exit psus and temples'
```

In [18]:

```
def find_len(txt):
    return len(txt.split())
```

In [19]:

```
df['Txt_len'] = [find_len(txt) for txt in df['clean_text']]
```

In [20]:

```
df.head()
```

Out[20]:

	clean_text	category	Txt_len
0	when modi promised “minimum government maximum...	Negative	33
1	talk all the nonsense and continue all the dra...	Neutral	13
2	what did just say vote for modi welcome bjp t...	Positive	22
3	asking his supporters prefix chowkidar their n...	Positive	34
4	answer who among these the most powerful world...	Positive	14

In [22]:

```
voc_size=5000
```

In [23]:

```
X = df.drop(["category","Txt_len"],axis = 1)
y = df.category
```

In [24]:

```
messages=X.copy()
```

In [25]:

```
messages['clean_text'][0]
```

Out[25]:

```
'when modi promised “minimum government maximum governance” expected him begin the dif
ficult job reforming the state why does take years get justice state should and not bu
siness and should exit psus and temples'
```

In [26]:

```
messages.reset_index(inplace=True)
```

In [27]:

```
import nltk
import re
from nltk.corpus import stopwords
```

In [28]:

```
from nltk.stem.porter import PorterStemmer
ps = PorterStemmer()
corpus = []
for i in range(0, len(messages)):
    print(i)
    review = re.sub('[^a-zA-Z]', ' ', messages['clean_text'][i])
    review = review.lower()
    review = review.split()

    review = [ps.stem(word) for word in review if not word in stopwords.words('english')]
    review = ' '.join(review)
    corpus.append(review)
```

```
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
```

In [29]:

```
corpus
```

```
 'crush jaw shoutmodimodi say jd mla incit murder',
 'sultanpur uttar pradesh loksabha candid select pawan kumar pandey actual public w
ant given vote modi current condid popular district candid bsp candid sonbhadra sin
gh',
 'thiugh nehru aliv still aliv heart modi everi failur nehru respons',
 'develop becom mass movement modi govt econom social polit empower life one wit po
sit paradigm shift new india',
 'alreadi taken notic order probe time modi take notic muslim famili harass beaten
recent extremist hindu suggest leav india move pakistan',
 'wait modi also talk varanasi',
 'accord yogi imran masood kin azhar masood accord logic nirav modi lalit modi nare
ndra modi brother mother',
 'agre tenur modiganga rejuven work start work',
 'three code modi crack give india huge foreign polici jumpstart via',
 'vote ensur govt need deserv anupam kher respond modi vote kar appeal elect',
 'modi govt slash india educ budget clear indic dont care india futur congress pres
id shri hand ensur increas budget gdp futur india deserv',
 'born religion femal deiti worship misogynist sadist tradit total point isit man m
ade tradit written one religi lunat support religion repress',
 'peopl made amazedn fear frustat may result vote sir wast ministerdisgrac entir mo
```

In [30]:

```
from tensorflow.keras.preprocessing.text import one_hot
onehot_repr=[one_hot(words,voc_size)for words in corpus]
onehot_repr
```

```
 [48, 2829, 1283, 3641, 1201, 3615],
 [1005, 1201, 3615, 3444, 1853, 1846, 664, 810, 4321, 3615, 1997, 3615, 3200],
 [3780,
  2267,
  3370,
  3234,
  4616,
  3615,
  2976,
  972,
  4036,
  4958,
  869,
  3081,
  3644,
  93,
  2829,
  4221,
  4916,
  4418,
```

In [31]:

```
from tensorflow.keras.preprocessing.sequence import pad_sequences
sent_length=20 # sentance length
embedded_docs=pad_sequences(onehot_repr,padding='pre',maxlen=sent_length)
print(embedded_docs)
```

```
[[1614  115 2371 ... 2998 4580 1364]
 [   0    0    0 ... 3641 1201 3615]
 [   0    0    0 ... 1997 3615 3200]
 ...
 [   0    0    0 ...  434 2699  413]
 [   0    0    0 ...  750 2908 1667]
 [3938 2925 2744 ... 1879 1148 1314]]
```

In [32]:

```
embedded_docs[0]
```

Out[32]:

```
array([1614,  115, 2371,  141, 2371, 2894, 2035,  865,  385,  757, 3802,
       1879, 3120, 4048,  620, 3802,  700, 2998, 4580, 1364])
```

In [33]:

```
from tensorflow.keras.layers import Embedding
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Bidirectional
from tensorflow.keras.layers import Dropout
```

In [34]:

```
embedding_vector_features=40
model1=Sequential()
model1.add(Embedding(voc_size,embedding_vector_features,input_length=sent_length))
model1.add(Bidirectional(LSTM(100)))
model1.add(Dropout(0.3))
model1.add(Dense(3,activation='softmax'))
model1.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
print(model1.summary())
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 20, 40)            200000

 bidirectional (Bidirectiona  (None, 200)              112800
 l)

 dropout (Dropout)           (None, 200)               0

 dense (Dense)               (None, 3)                 603

=================================================================
Total params: 313,403
Trainable params: 313,403
Non-trainable params: 0
_________________________________________________________________
None
```

In [35]:

```
len(embedded_docs),y.shape
```

Out[35]:

```
(162969, (162969,))
```

In [36]:

```
import numpy as np
X_final=np.array(embedded_docs)
y_final=np.array(y)
```

In [37]:

```
X_final.shape,y_final.shape
```

Out[37]:

```
((162969, 20), (162969,))
```

In [38]:

```
y_final
```

Out[38]:

```
array(['Negative', 'Neutral', 'Positive', ..., 'Neutral', 'Neutral',
       'Positive'], dtype=object)
```

In [39]:

```
y_final = pd.get_dummies(y_final)
y_final
```

Out[39]:

	Negative	Neutral	Positive
0	1	0	0
1	0	1	0
2	0	0	1
3	0	0	1
4	0	0	1
...	...	...	...
162964	1	0	0
162965	1	0	0
162966	0	1	0
162967	0	1	0
162968	0	0	1

162969 rows × 3 columns

In [40]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_final, y_final, test_size=0.2, random_state=42
```

In [41]:

```
model1.fit(X_train,y_train, validation_data=(X_test,y_test),epochs=10,batch_size=64)
```

```
Epoch 1/10
2038/2038 [==============================] - 58s 26ms/step - loss: 0.3896 - accuracy:
0.7368 - val_loss: 0.3474 - val_accuracy: 0.7754
Epoch 2/10
2038/2038 [==============================] - 48s 24ms/step - loss: 0.3366 - accuracy:
0.7830 - val_loss: 0.3442 - val_accuracy: 0.7760
Epoch 3/10
2038/2038 [==============================] - 46s 23ms/step - loss: 0.3222 - accuracy:
0.7920 - val_loss: 0.3375 - val_accuracy: 0.7778
Epoch 4/10
2038/2038 [==============================] - 44s 22ms/step - loss: 0.3073 - accuracy:
0.8031 - val_loss: 0.3426 - val_accuracy: 0.7765
Epoch 5/10
2038/2038 [==============================] - 49s 24ms/step - loss: 0.2913 - accuracy:
0.8138 - val_loss: 0.3483 - val_accuracy: 0.7770
Epoch 6/10
2038/2038 [==============================] - 56s 27ms/step - loss: 0.2755 - accuracy:
0.8259 - val_loss: 0.3533 - val_accuracy: 0.7749
Epoch 7/10
2038/2038 [==============================] - 56s 28ms/step - loss: 0.2588 - accuracy:
0.8379 - val_loss: 0.3688 - val_accuracy: 0.7693
Epoch 8/10
2038/2038 [==============================] - 60s 29ms/step - loss: 0.2425 - accuracy:
0.8506 - val_loss: 0.3924 - val_accuracy: 0.7680
Epoch 9/10
2038/2038 [==============================] - 62s 30ms/step - loss: 0.2252 - accuracy:
0.8623 - val_loss: 0.4107 - val_accuracy: 0.7659
Epoch 10/10
2038/2038 [==============================] - 61s 30ms/step - loss: 0.2088 - accuracy:
0.8740 - val_loss: 0.4423 - val_accuracy: 0.7603
```

Out[41]:

```
<keras.callbacks.History at 0x16fd7922430>
```

In [42]:

```
y_pred1 = model1.predict(X_test)
```

```
1019/1019 [==============================] - 9s 5ms/step
```

In [43]:

```
y_pred1[0]
```

Out[43]:

```
array([6.6097600e-05, 9.9962449e-01, 3.0929982e-04], dtype=float32)
```

In [44]:

```
y_pred1[0:10]
```

Out[44]:

```
array([[6.6097600e-05, 9.9962449e-01, 3.0929982e-04],
       [5.8121810e-04, 9.8850584e-01, 1.0912958e-02],
       [9.0695202e-01, 4.2134650e-02, 5.0913353e-02],
       [5.8696564e-02, 1.9338989e-01, 7.4791354e-01],
       [3.0866444e-01, 3.7603030e-01, 3.1530529e-01],
       [9.9789512e-01, 2.6274612e-04, 1.8420593e-03],
       [7.6944619e-02, 8.7783724e-01, 4.5218185e-02],
       [6.8862224e-01, 6.0366564e-03, 3.0534112e-01],
       [1.2742368e-05, 2.2973727e-06, 9.9998498e-01],
       [7.9907186e-02, 4.4492446e-03, 9.1564357e-01]], dtype=float32)
```

In [45]:

```
ex = y_pred1.copy()
ex[1]
```

Out[45]:

```
array([5.8121810e-04, 9.8850584e-01, 1.0912958e-02], dtype=float32)
```

In [46]:

```
count = 0
for i in range(len(ex)):
    if (ex[i][0] > ex[i][1] and ex[i][0] > ex[i][2]):
        count = 0

    elif(ex[i][0] < ex[i][1] and ex[i][1] > ex[i][2]):
        count = 1

    elif(ex[i][0] < ex[i][2] and ex[i][1] < ex[i][2]):
        count = 2
    print(i,"for count is",count)
    for j in range(3):
        ex[i][j] = 0
        ex[i][count] = 1
    print(ex[i])
    count = 0
```

```
0 for count is 1
[0. 1. 0.]
1 for count is 1
[0. 1. 0.]
2 for count is 0
[1. 0. 0.]
3 for count is 2
[0. 0. 1.]
4 for count is 1
[0. 1. 0.]
5 for count is 0
[1. 0. 0.]
6 for count is 1
[0. 1. 0.]
7 for count is 0
[1. 0. 0.]
8 for count is 2
[0. 0. 1.]
9 for count is 2
```

In [47]:

```
y_test
```

Out[47]:

	Negative	Neutral	Positive
42228	0	1	0
22034	0	0	1
79981	1	0	0
118492	1	0	0
12814	0	1	0
...	...	...	...
47104	0	0	1
33631	1	0	0
93675	0	1	0
37756	0	1	0
132995	0	0	1

32594 rows × 3 columns

In [48]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,ex)
```

Out[48]:

```
0.7602933055163527
```

In [49]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test,ex))
```

```
              precision    recall  f1-score   support

           0       0.66      0.63      0.65      7152
           1       0.78      0.79      0.79     11067
           2       0.79      0.80      0.79     14375

   micro avg       0.76      0.76      0.76     32594
   macro avg       0.74      0.74      0.74     32594
weighted avg       0.76      0.76      0.76     32594
 samples avg       0.76      0.76      0.76     32594
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

```
```