Import the all necessary library

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
import string
import nltk
import warnings
%matplotlib inline

warnings.filterwarnings('ignore')
```

Version of GPU

```
gpu_info = !nvidia-smi
gpu_info = '\n'.join(gpu_info)
if gpu_info.find('failed') >= 0:
  print('Not connected to a GPU')
else:
  print(gpu_info)
```

```
Thu Jul  4 15:29:38 2024
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-----------------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|=========================================+======================+======================|
|   0  NVIDIA L4                      Off | 00000000:00:03.0 Off |                    0 |
| N/A   40C    P8              12W /  72W |      1MiB / 23034MiB |      0%      Default |
|                                         |                      |                  N/A |
+-----------------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI        PID   Type   Process name                            GPU Memory |
|        ID   ID                                                             Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
```

Import the Twitter Dataset

```
input_file = '/content/Twitter_Data.csv'
```

Converted Into DataFrame

```
input_df=pd.read_csv(input_file)
input_df.head()
```

	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

```
input_df
```

	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
...	...	...
162975	why these 456 crores paid neerav modi not reco...	-1.0
162976	dear rss terrorist payal gawar what about modi...	-1.0
162977	did you cover her interaction forum where she ...	0.0
162978	there big project came into india modi dream p...	0.0
162979	have you ever listen about like gurukul where ...	1.0

162980 rows × 2 columns

find Shape of DataFrame

```
input_df.shape
```

```
(162980, 2)
```

check the Null values present in Data set

```
input_df.isnull().sum()
```

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

Remove null value

```
input_df.dropna(inplace=True)
```

```
input_df.isnull().sum()
```

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

```
input_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 162969 entries, 0 to 162979
Data columns (total 2 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   clean_text  162969 non-null  object
 1   category    162969 non-null  float64
dtypes: float64(1), object(1)
memory usage: 3.7+ MB
```

Our category column present Float value converted into int

```
input_df['category'] =input_df['category'].astype(int)
```

change the label

- List item positive(+1)--->postive(2)
- List item Negative(-1)--->Negative(0)
- List item Neutral(0)----->neutral(1)

```
input_df['category'] = input_df['category'].replace({-1: 0, 0: 1, 1: 2})
```

```
input_df
```

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

162969 rows × 2 columns

```
null_check = input_df.isnull()
print(null_check.sum())
```

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

```
input_df.describe()
```

	category
count	162969.000000
mean	1.225442
std	0.781279
min	0.000000
25%	1.000000
50%	1.000000
75%	2.000000
max	2.000000

check length of each clean_text

```
input_df['clean_text'].apply(len)
```

```
0         210
1          68
2         117
3         212
4          81
         ...
162975    108
162976    248
162977     51
162978     77
162979    216
Name: clean_text, Length: 162969, dtype: int64
```

below the and above 5 number clean_text in data set

```
sum(input_df['clean_text'].apply(len)>5),sum(input_df['clean_text'].apply(len)<5)
```

```
(162870, 43)
```

Keep bigger then 5 word in clean_text

```
input_df=input_df[input_df['clean_text'].apply(len)>5]
print(len(input_df))
```

```
162870
```

count the number of Label present in each category

```
input_df['category'].value_counts()
```

```
category
2    72230
1    55132
0    35508
Name: count, dtype: int64
```

```
def remove_pattern(input_txt, pattern):
    r = re.findall(pattern, input_txt)
    for word in r:
        input_txt = re.sub(word, "", input_txt)
    return input_txt
```

```
input_df.head()
```

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

check word start with the @ because start of tweet everyone used @

```
import re
def count_handles(text):
    return len(re.findall(r'@[\w]+', text))

input_df['handle_count'] = input_df['clean_text'].apply(count_handles)

input_df
```

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

162870 rows × 3 columns

n0 @ found in any row

Check how special character,punchuation and numeric present in clean_text

```
def count_special_characters(text):
    special_chars = re.findall(r'[^a-zA-Z0-9\s]', text)
    return len(special_chars)

def count_punctuation(text):
    punctuation = re.findall(r'[^\w\s]', text)
    return len(punctuation)

def count_numbers(text):
    numbers = re.findall(r'[0-9]', text)
    return len(numbers)

input_df['special_char_count'] = input_df['clean_text'].apply(count_special_characters)
input_df['punctuation_count'] = input_df['clean_text'].apply(count_punctuation)
input_df['number_count'] = input_df['clean_text'].apply(count_numbers)

input_df
```

	clean_text	category	handle_count	special_char_count	punctuation_count	numb
0	when modi promised “minimum government maximum...	0	0	2	2	
1	talk all the nonsense and continue all the dra...	1	0	0	0	
2	what did just say vote for modi welcome bjp t...	2	0	0	0	
	asking his supporters					

count special character,punchuation and numeric present in clean_text

```
total_handle_count = input_df['handle_count'].sum()
total_special_char_count = input_df['special_char_count'].sum()
total_punctuation_count = input_df['punctuation_count'].sum()
total_number_count = input_df['number_count'].sum()

print(f"Total Handle Count: {total_handle_count}")
print(f"Total Special Character Count: {total_special_char_count}")
print(f"Total Punctuation Count: {total_punctuation_count}")
print(f"Total Number Count: {total_number_count}")
```

```
Total Handle Count: 0
Total Special Character Count: 47399
Total Punctuation Count: 44266
Total Number Count: 123969
```

Now remove all special character,punchuation and numeric

```
def remove_special_punctuation_numbers(text):
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    return text

input_df['clean_text_more'] = input_df['clean_text'].apply(remove_special_punctuation_numbers)

input_df
```

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

```
input_df['clean_tweet_'] = input_df['clean_text_more'].apply(lambda x: " ".join([w for w in x.split() if len(w)>3]))
input_df.head()
```

	clean_text	category	handle_count	special_char_count	punctuation_count	number_co
0	when modi promised “minimum government maximum...	0	0	2	2	
1	talk all the nonsense and continue all the dra...	1	0	0	0	
2	what did just say vote for modi welcome bjp t...	2	0	0	0	
	asking his					

**Drop some extra column **

```
input_df = input_df.drop(['handle_count','special_char_count','punctuation_count','number_count'], axis=1)

input_df
```

	clean_text	category	clean_text_more	clean_tweet_
0	when modi promised “minimum government maximum...	0	when modi promised minimum government maximum ...	when modi promised minimum government maximum ...
1	talk all the nonsense and continue all the dra...	1	talk all the nonsense and continue all the dra...	talk nonsense continue drama will vote modi
2	what did just say vote for modi welcome bjp t...	2	what did just say vote for modi welcome bjp t...	what just vote modi welcome told rahul main ca...
3	asking his supporters prefix chowkidar their n...	2	asking his supporters prefix chowkidar their n...	asking supporters prefix chowkidar their names...
4	answer who among these the most powerful world...	2	answer who among these the most powerful world...	answer among these most powerful world leader ...
...	...	...	...	...
162975	why these 456 crores paid neerav modi not reco...	0	why these crores paid neerav modi not recover...	these crores paid neerav modi recovered from c...

```
input_df.columns
```

```
Index(['clean_text', 'category', 'clean_text_more', 'clean_tweet_'], dtype='object')
```

Stemming apply like running---> run

```
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
```

```
stemmer = PorterStemmer()
```

```
def stem_text(text):
    words = word_tokenize(text)
    stemmed_words = [stemmer.stem(word) for word in words]
    return ' '.join(stemmed_words)
```

```
import nltk
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize

nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
input_df['clean_tweet_'] = input_df['clean_tweet_'].apply(stem_text)
```

```
input_df
```

	clean_text	category	clean_text_more	clean_tweet_
0	when modi promised “minimum government maximum...	0	when modi promised minimum government maximum ...	when modi promis minimum govern maximum govern...
1	talk all the nonsense and continue all the dra...	1	talk all the nonsense and continue all the dra...	talk nonsens continu drama will vote modi
2	what did just say vote for modi welcome bjp t...	2	what did just say vote for modi welcome bjp t...	what just vote modi welcom told rahul main cam...
3	asking his supporters prefix chowkidar their n...	2	asking his supporters prefix chowkidar their n...	ask support prefix chowkidar their name modi g...
4	answer who among these the most powerful world...	2	answer who among these the most powerful world...	answer among these most power world leader tod...
...	...	...	...	...
162975	why these 456 crores paid neerav modi not reco...	0	why these crores paid neerav modi not recover...	these crore paid neerav modi recov from congre...
162976	dear rss terrorist payal gawar what about	0	dear rss terrorist payal gawar what about modi	dear terrorist payal gawar what about modi

load in excel file and applied to BERT Prediction

```
import pandas as pd
output_file = '/content/modified_input_df.xlsx'
input_df.to_excel(output_file, index=False)

from google.colab import files
files.download(output_file)
```

```
!pip install wordcloud
```

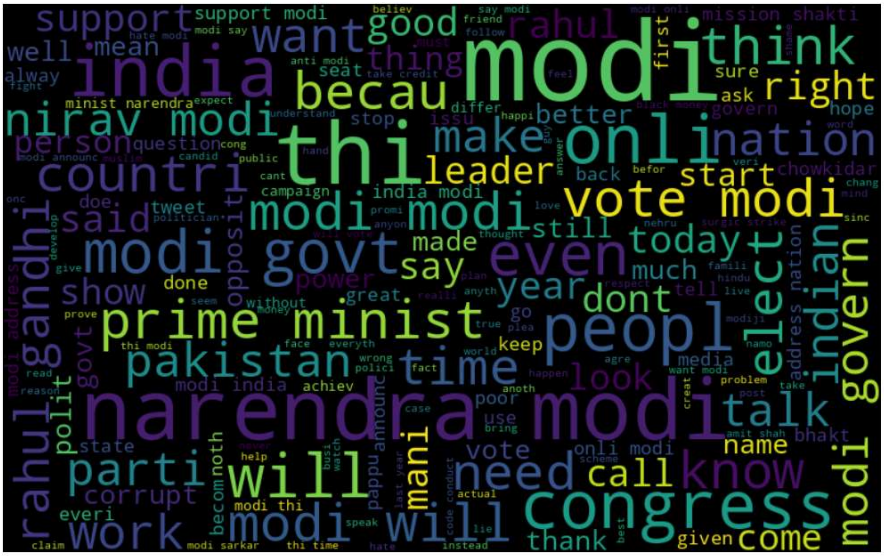
```
Requirement already satisfied: wordcloud in /usr/local/lib/python3.10/dist-packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in /usr/local/lib/python3.10/dist-packages (from wordcloud) (1.25.2)
Requirement already satisfied: pillow in /usr/local/lib/python3.10/dist-packages (from wordcloud) (9.4.0)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from wordcloud) (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (1.4.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (24.1)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->wordcloud) (
```

```
input_df.isnull().sum()
```

```
clean_text         0
category           0
clean_text_more    0
clean_tweet_       0
dtype: int64
```

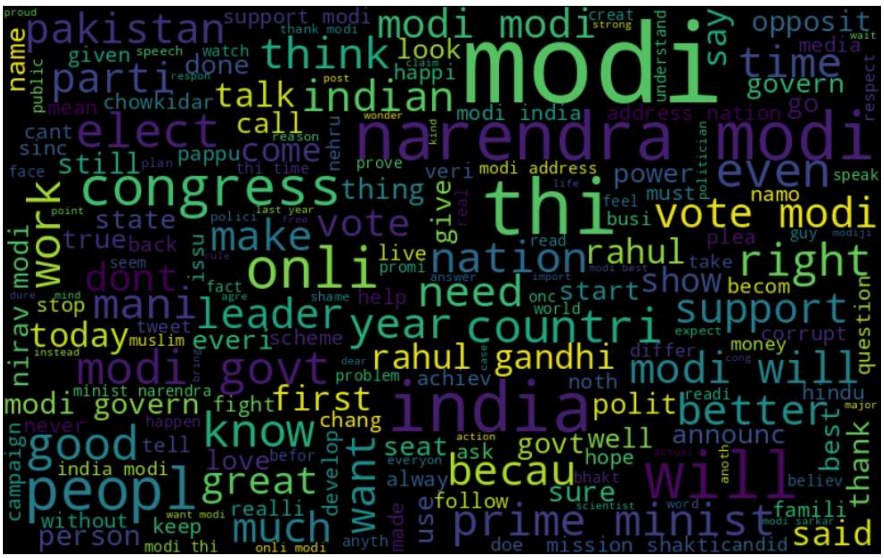
frequecy of word is hight then word look more bold

```
all_words = " ".join([sentence for sentence in input_df['clean_tweet_']])

from wordcloud import WordCloud
wordcloud = WordCloud(width=800, height=500, random_state=42, max_font_size=100).generate(all_words)

plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



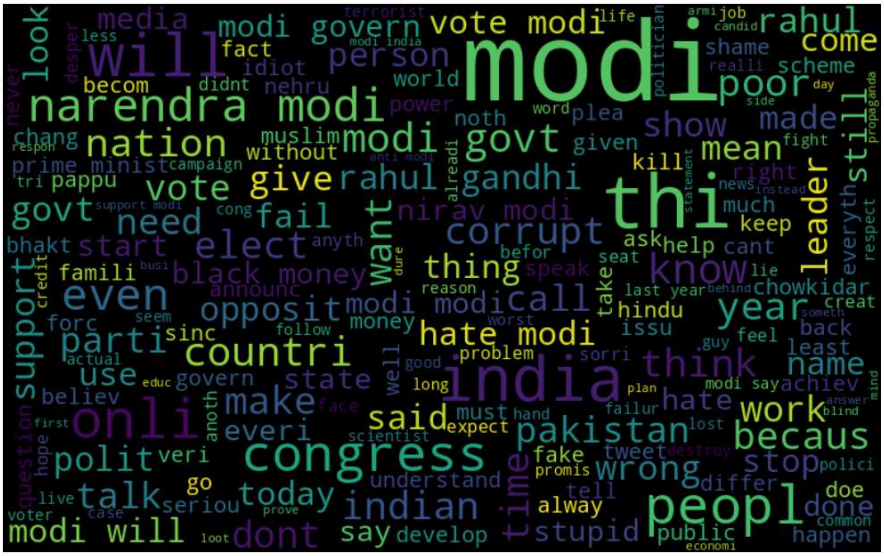
frequent words visualization for +ve

```
all_words = " ".join([sentence for sentence in input_df['clean_tweet_'][input_df['category']==2]])

wordcloud = WordCloud(width=800, height=500, random_state=42, max_font_size=100).generate(all_words)

plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



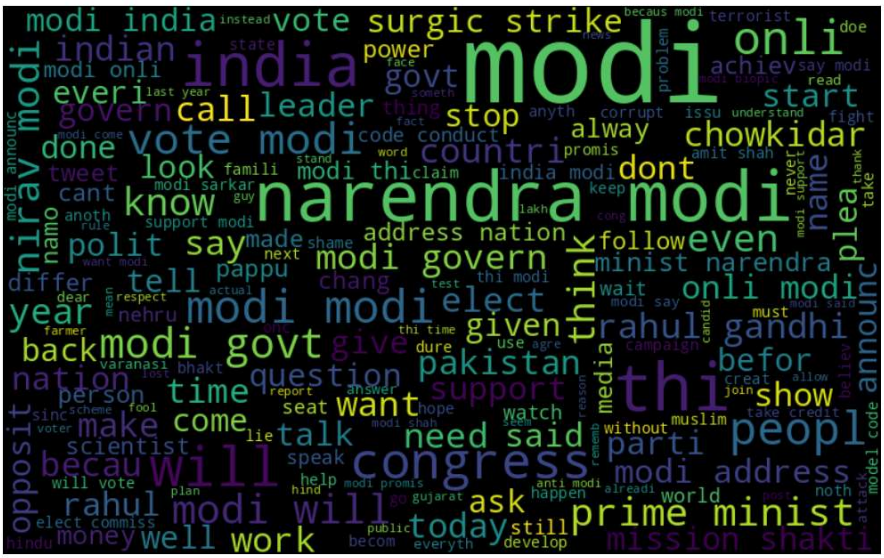
frequent words visualization for -ive

```
all_words = " ".join([sentence for sentence in input_df['clean_tweet_'][input_df['category']==0]])

wordcloud = WordCloud(width=800, height=500, random_state=42, max_font_size=100).generate(all_words)

plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



frequent words visualization for nuetral

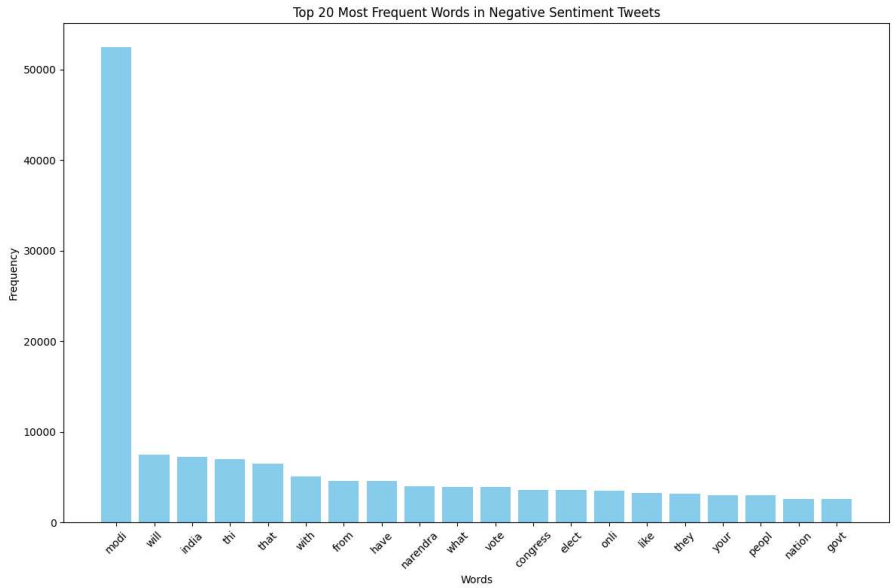
```
all_words = " ".join([sentence for sentence in input_df['clean_tweet_'][input_df['category']==1]])

wordcloud = WordCloud(width=800, height=500, random_state=42, max_font_size=100).generate(all_words)

plt.figure(figsize=(15,8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```



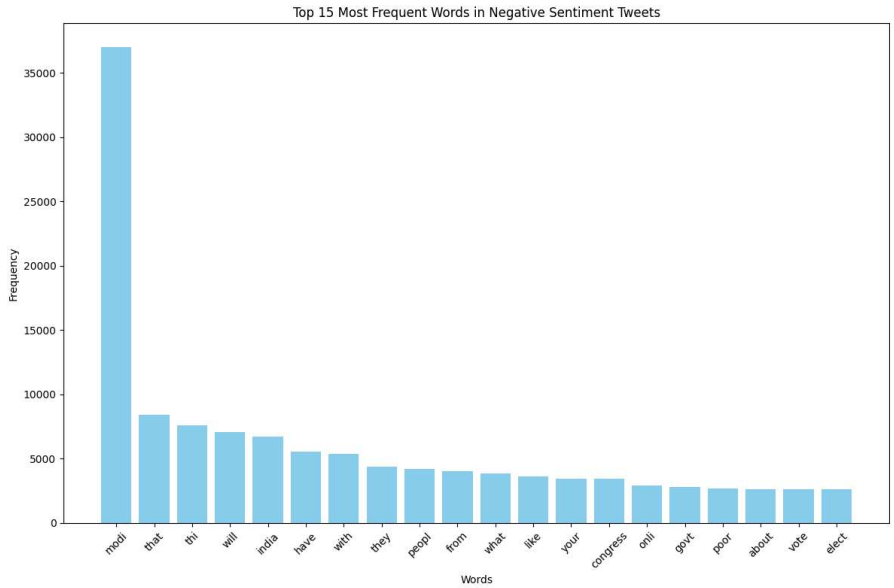

```
import matplotlib.pyplot as plt
from collections import Counter
from wordcloud import WordCloud
```

Concatenate all 'clean_tweet_' texts where category is neutral

```
import matplotlib.pyplot as plt
from collections import Counter
all_words = " ".join(input_df[input_df['category'] == 1]['clean_tweet_'])
words = all_words.split()
word_freq = Counter(words)
top_words = word_freq.most_common(20)
top_words, frequencies = zip(*top_words)
plt.figure(figsize=(12, 8))
plt.bar(top_words, frequencies, color='skyblue')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Top 20 Most Frequent Words in Negative Sentiment Tweets')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

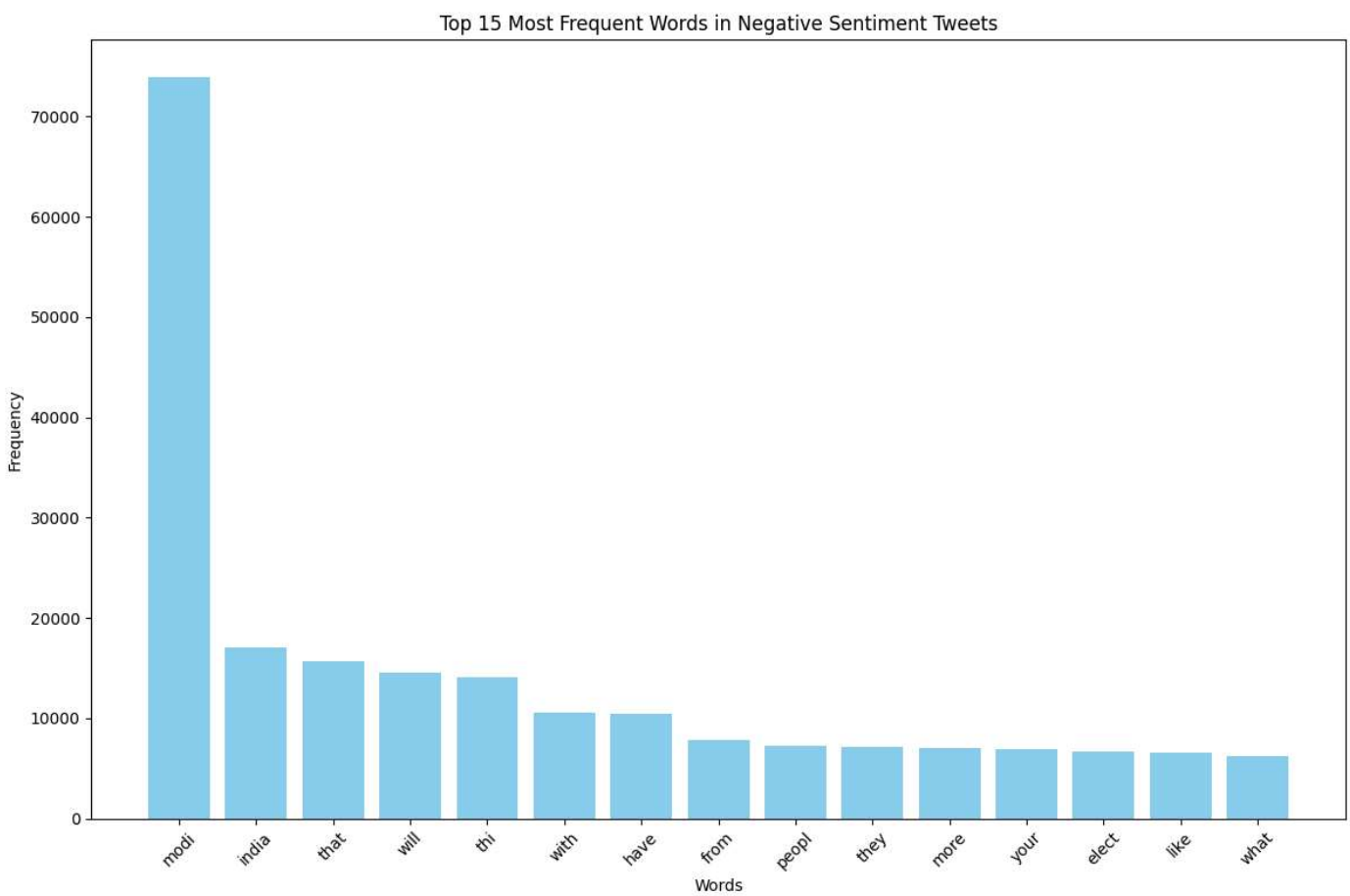



```
all_words = " ".join(input_df[input_df['category'] == 0]['clean_tweet_'])
words = all_words.split()
word_freq = Counter(words)
top_words = word_freq.most_common(20)
top_words, frequencies = zip(*top_words)
plt.figure(figsize=(12, 8))
plt.bar(top_words, frequencies, color='skyblue')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Top 15 Most Frequent Words in Negative Sentiment Tweets')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Concatenate all 'clean_tweet_' texts where category is +ive

```
all_words = " ".join(input_df[input_df['category'] == 2]['clean_tweet_'])
words = all_words.split()
word_freq = Counter(words)
top_words = word_freq.most_common(15)
top_words, frequencies = zip(*top_words)
plt.figure(figsize=(12, 8))
plt.bar(top_words, frequencies, color='skyblue')
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Top 15 Most Frequent Words in Negative Sentiment Tweets')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Double-click (or enter) to edit

```
from sklearn.feature_extraction.text import CountVectorizer
bow_vectorizer = CountVectorizer(max_df=0.90, min_df=2, max_features=100000, stop_words='english')
bow = bow_vectorizer.fit_transform(input_df['clean_tweet_'])
```

splitting the data set train and test

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(bow, input_df['category'], random_state=42, test_size=0.25)
```

LogisticRegression method is used for training and Testing

1. Bag of Words (Count Vectorizer)
2. TF-IDF Vectorizer
3. Word2Vec
4. FastText

Logistic regression Method Use for training and Testing

```
import pandas as pd
import numpy as np
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from gensim.models import Word2Vec, FastText
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Assuming input_df and tokens are already defined as in your previous code

vectorizations = {}

# 1. Bag of Words (Count Vectorizer)
bow_vectorizer = CountVectorizer(max_df=0.90, min_df=2, max_features=100000, stop_words='english')
vectorizations['bow'] = bow_vectorizer.fit_transform(input_df['clean_tweet_'])

# 2. TF-IDF Vectorizer
tfidf_vectorizer = TfidfVectorizer(max_df=0.90, min_df=2, max_features=100000, stop_words='english')
vectorizations['tfidf'] = tfidf_vectorizer.fit_transform(input_df['clean_tweet_'])

# 3. Word2Vec
word2vec_model = Word2Vec(sentences=tokens, vector_size=100, window=5, min_count=2, workers=4)
word2vec_vectors = [np.mean([word2vec_model.wv[word] for word in words if word in word2vec_model.wv] or [np.zeros(100)], axis=0) for words
vectorizations['word2vec'] = np.array(word2vec_vectors)

# 4. FastText
fasttext_model = FastText(sentences=tokens, vector_size=100, window=5, min_count=2, workers=4)
fasttext_vectors = [np.mean([fasttext_model.wv[word] for word in words if word in fasttext_model.wv] or [np.zeros(100)], axis=0) for words
vectorizations['fasttext'] = np.array(fasttext_vectors)

# Define classifiers
logreg = LogisticRegression(max_iter=1000)
classifiers = [('Logistic Regression', logreg)]

# Train and evaluate each vectorization method with each classifier
for vec_name, vec_data in vectorizations.items():
    print(f"Vectorization: {vec_name}")
    x_train, x_test, y_train, y_test = train_test_split(vec_data, input_df['category'], random_state=42, test_size=0.25)

    for clf_name, clf in classifiers:
        clf.fit(x_train, y_train)
        pred = clf.predict(x_test)

        # Calculate evaluation metrics
        accuracy = accuracy_score(y_test, pred)
        precision = precision_score(y_test, pred, average='macro')
        recall = recall_score(y_test, pred, average='macro')
```