```
import pandas as pd

path="/content/drive/MyDrive/Twitter Sentiments.csv"

df=pd.read_csv(path)

df.head()
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

```
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')

path="/content/drive/MyDrive/Twitter Sentiments.csv"

df1=pd.read_csv(path)
df1.head()
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

```
# datatype info
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31962 entries, 0 to 31961
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      31962 non-null  int64 
 1   label   31962 non-null  int64 
 2   tweet   31962 non-null  object
dtypes: int64(2), object(1)
memory usage: 749.2+ KB
```

```
# removes pattern in the input text
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
```

```
df1.head()
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

```
# remove twitter handles (@user)
df1['clean_tweet'] = np.vectorize(remove_pattern)(df1['tweet'], "@[\w]*")
```

```
df1.head()
```

	id	label	tweet	clean_tweet
0	1	0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can't use cause th...
2	3	0	bihday your majesty	bihday your majesty
3	4	0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...

```
# remove special characters, numbers and punctuations
df1['clean_tweet'] = df1['clean_tweet'].str.replace("[^a-zA-Z#]", " ")
df1.head()
```

	id	label	tweet	clean_tweet
0	1	0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can t use cause th...
2	3	0	bihday your majesty	bihday your majesty
3	4	0	#model i love u take with u all the time in	#model i love u take with u all the time

```
# remove short words
df1['clean_tweet'] = df1['clean_tweet'].apply(lambda x: " ".join([w for w in x.split() if
df1.head()
```

	id	label	tweet	clean_tweet
0	1	0	@user when a father is dysfunctional and is s...	when father dysfunctional selfish drags kids i...
1	2	0	@user @user thanks for #lyft credit i can't us...	thanks #lyft credit cause they offer wheelchai...
2	3	0	bihday your majesty	bihday your majesty
3	4	0	#model i love u take with u all the time in ...	#model love take with time

```
# individual words considered as tokens
tokenized_tweet = df1['clean_tweet'].apply(lambda x: x.split())
tokenized_tweet.head()
```

```
0    [when, father, dysfunctional, selfish, drags, ...
1    [thanks, #lyft, credit, cause, they, offer, wh...
2                               [bihday, your, majesty]
3                    [#model, love, take, with, time]
4                  [factsguide, society, #motivation]
Name: clean_tweet, dtype: object
```

```
# stem the words
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()

tokenized_tweet = tokenized_tweet.apply(lambda sentence: [stemmer.stem(word) for word in s
tokenized_tweet.head()
```

```
0    [when, father, dysfunct, selfish, drag, kid, i...
1    [thank, #lyft, credit, caus, they, offer, whee...
2                               [bihday, your, majesti]
3                    [#model, love, take, with, time]
4                        [factsguid, societi, #motiv]
Name: clean_tweet, dtype: object
```

```
# combine words into single sentence
for i in range(len(tokenized_tweet)):
    tokenized_tweet[i] = " ".join(tokenized_tweet[i])
```

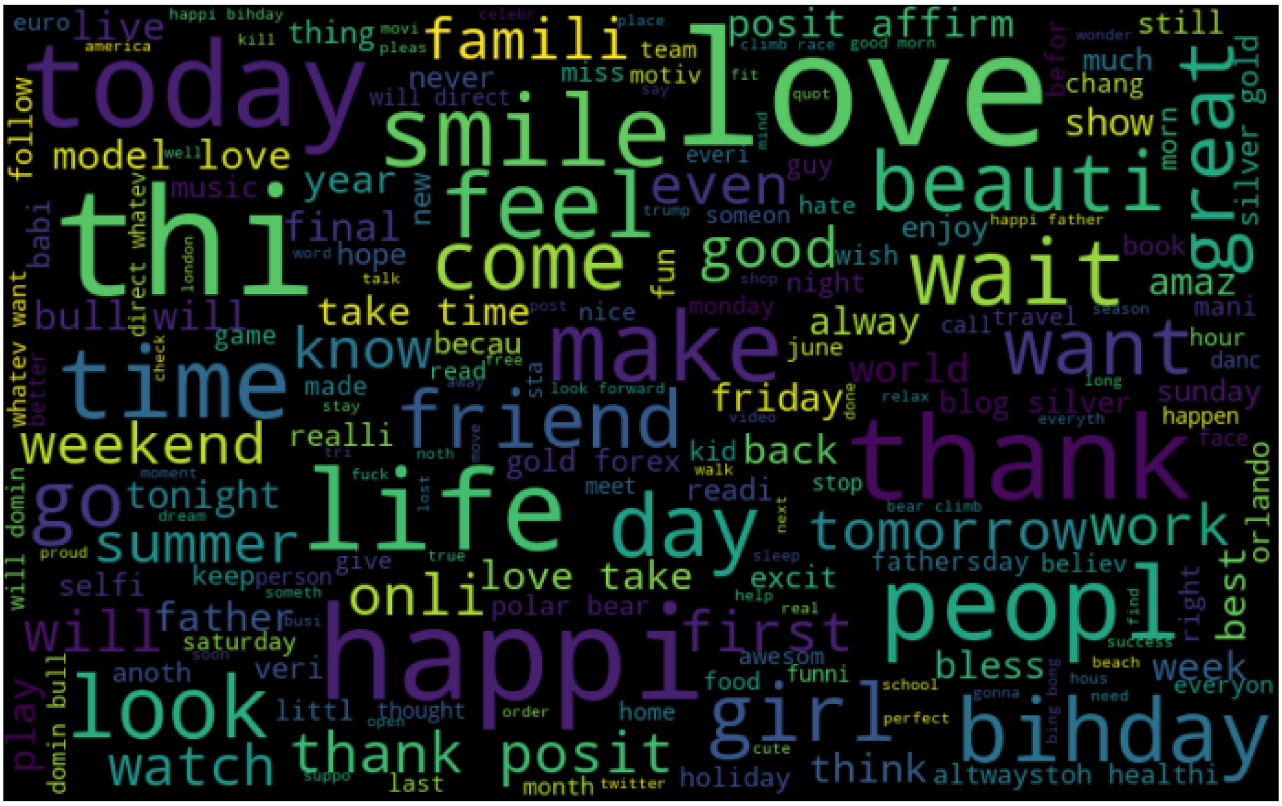
```
df1['clean_tweet'] = tokenized_tweet
df1.head()
```

	id	label	tweet	clean_tweet
0	1	0	@user when a father is dysfunctional and is s...	when father dysfunct selfish drag kid into dys...
1	2	0	@user @user thanks for #lyft credit i can't us...	thank #lyft credit caus they offer wheelchair ...
2	3	0	bihday your majesty	bihday your majesti
3	4	0	#model i love u take with u all the time in ...	#model love take with time

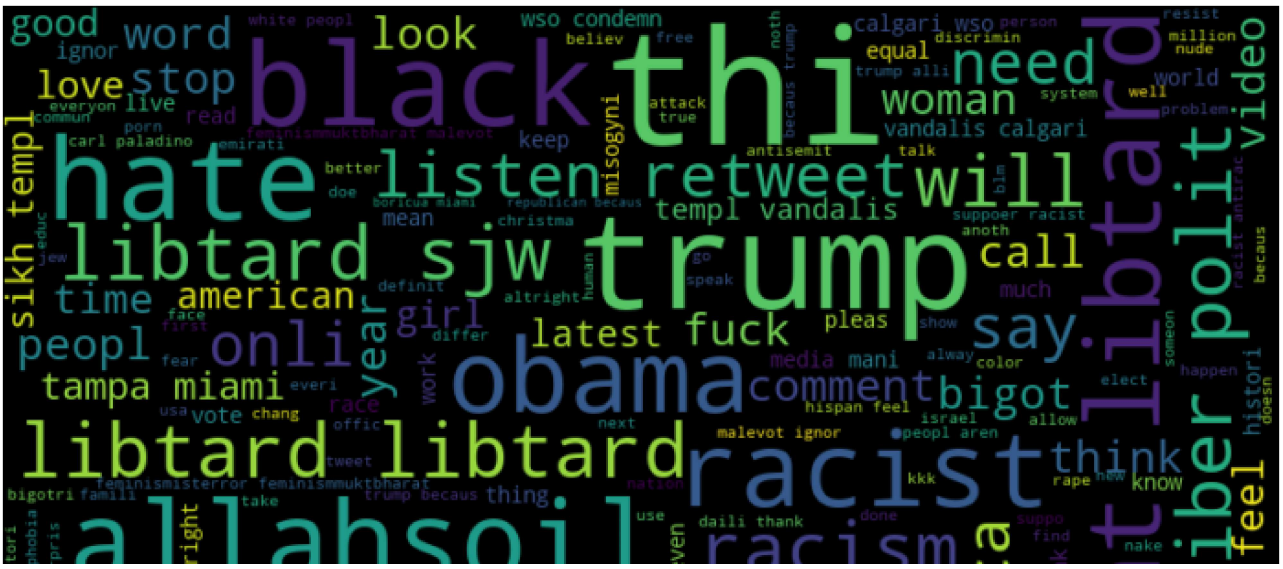
```
# visualize the frequent words
all_words = " ".join([sentence for sentence in df1['clean_tweet']])

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

# plot the graph
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 df1['clean_tweet'][df1['label']==0]])

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

# plot the graph
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 df1['clean_tweet'][df1['label']==1]])

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

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




```
# extract the hashtag
def hashtag_extract(tweets):
    hashtags = []
    # loop words in the tweet
    for tweet in tweets:
        ht = re.findall(r"#(\w+)", tweet)
        hashtags.append(ht)
    return hashtags


# extract hashtags from non-racist/sexist tweets
ht_positive = hashtag_extract(df1['clean_tweet'][df1['label']==0])

# extract hashtags from racist/sexist tweets
ht_negative = hashtag_extract(df1['clean_tweet'][df1['label']==1])


ht_positive[:5]
```

```
[['run'], ['lyft', 'disapoint', 'getthank'], [], ['model'], ['motiv']]
```

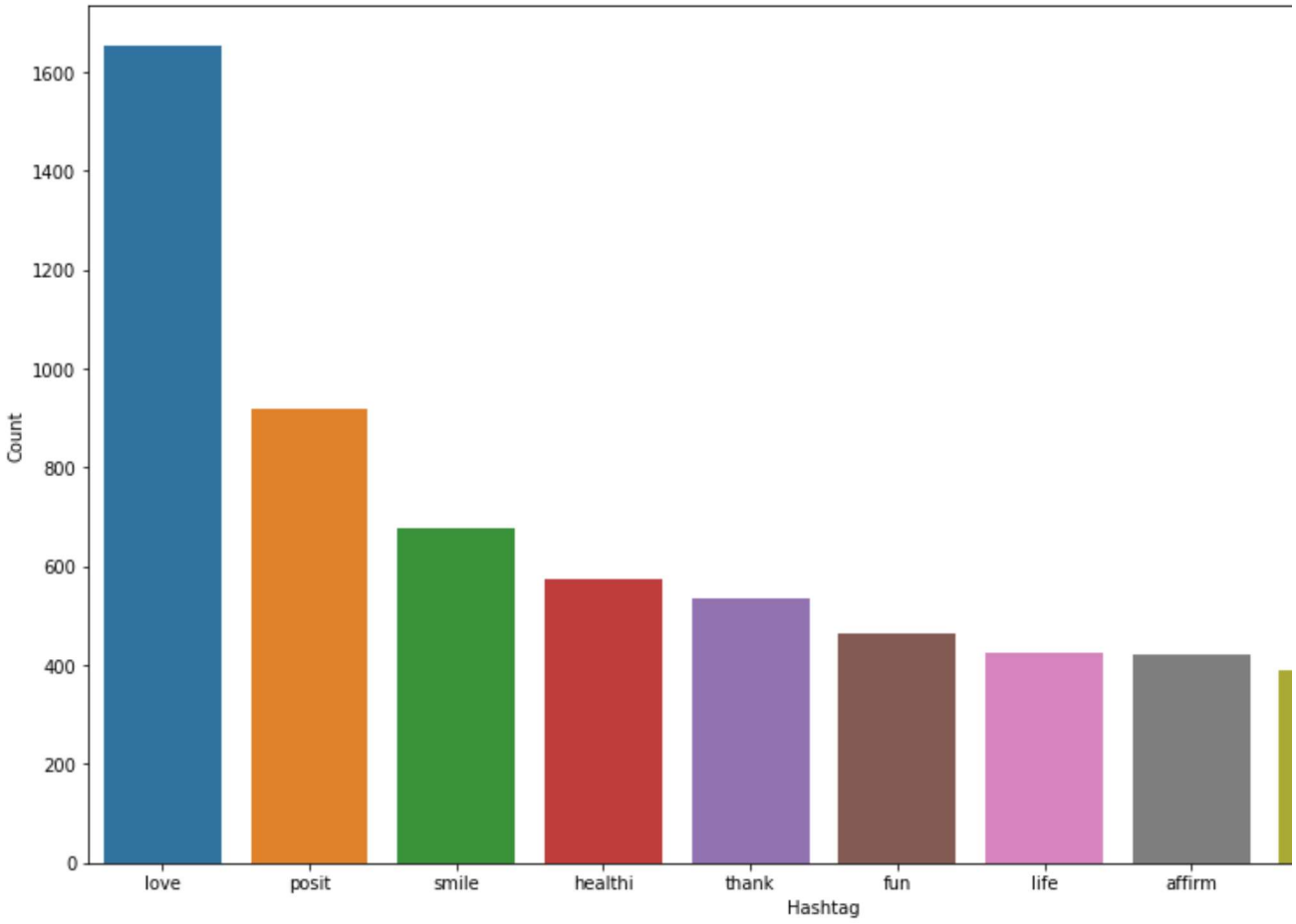
```
# unnest list
ht_positive = sum(ht_positive, [])
ht_negative = sum(ht_negative, [])


ht_positive[:5]
```

```
['run', 'lyft', 'disapoint', 'getthank', 'model']
```

```
freq = nltk.FreqDist(ht_positive)
d = pd.DataFrame({'Hashtag': list(freq.keys()),
                  'Count': list(freq.values())})
d.head()
```

	Hashtag	Count
0	run	72
1	lyft	2
2	disapoint	1

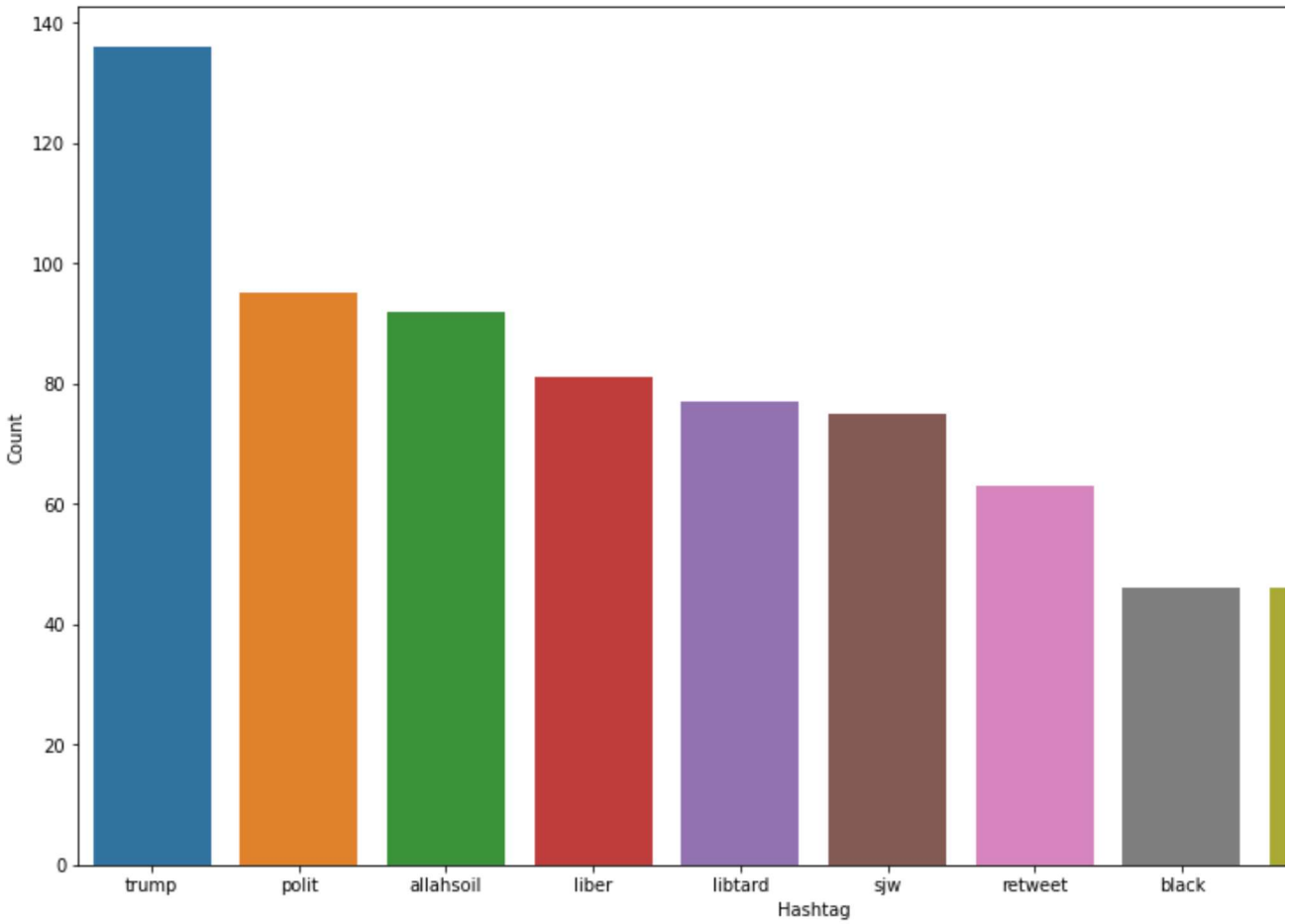
```
# select top 10 hashtags
d = d.nlargest(columns='Count', n=10)
plt.figure(figsize=(15,9))
sns.barplot(data=d, x='Hashtag', y='Count')
plt.show()
```




```
freq = nltk.FreqDist(ht_negative)
d = pd.DataFrame({'Hashtag': list(freq.keys()),
                  'Count': list(freq.values())})
d.head()
```

	Hashtag	Count
0	cnn	10

```
# select top 10 hashtags
d = d.nlargest(columns='Count', n=10)
plt.figure(figsize=(15,9))
sns.barplot(data=d, x='Hashtag', y='Count')
plt.show()
```




```
# feature extraction
from sklearn.feature_extraction.text import CountVectorizer
```

```
bow_vectorizer = CountVectorizer()
```

```
bow = bow_vectorizer.fit_transform(df1['clean_tweet'])
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(bow, df1['label'], random_state=42, te
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import f1_score, accuracy_score


# training
model = LogisticRegression()
model.fit(x_train, y_train)
```

```
LogisticRegression()
```

```
# testing
pred = model.predict(x_test)
f1_score(y_test, pred)
```

```
0.6359550561797752
```

```
accuracy_score(y_test,pred)
```

```
0.9594543861844576
```

```
# use probability to get output
pred_prob = model.predict_proba(x_test)
pred = pred_prob[:, 1] >= 0.3
pred = pred.astype(np.int)

f1_score(y_test, pred)
```

```
0.671206225680934
```

```
accuracy_score(y_test,pred)
```

```
0.9577024152171193
```

```
pred_prob[0][1] >= 0.3
```

```
False
```

✓ 0s completed at 12:14 AM