In [1]:
```
import numpy as np
import pandas as pd
```

In [3]:
```
data=pd.read_csv("IMDB Dataset.csv")
data
```

Out[3]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
...	...	...
49995	I thought this movie did a down right good job...	positive
49996	Bad plot, bad dialogue, bad acting, idiotic di...	negative
49997	I am a Catholic taught in parochial elementary...	negative
49998	I'm going to have to disagree with the previou...	negative
49999	No one expects the Star Trek movies to be high...	negative

50000 rows × 2 columns

In [7]:
```
df = data.iloc[:10000]
```

In [12]:
```
df['review'][1]
```

Out[12]:
```
'A wonderful little production. <br /><br />The filming technique is very unassumi
ng- very old-time-BBC fashion and gives a comforting, and sometimes discomforting,
sense of realism to the entire piece. <br /><br />The actors are extremely well ch
osen- Michael Sheen not only "has got all the polari" but he has all the voices do
wn pat too! You can truly see the seamless editing guided by the references to Wil
liams\' diary entries, not only is it well worth the watching but it is a terrific
ly written and performed piece. A masterful production about one of the great mast
er\'s of comedy and his life. <br /><br />The realism really comes home with the l
ittle things: the fantasy of the guard which, rather than use the traditional \'dr
eam\' techniques remains solid then disappears. It plays on our knowledge and our
senses, particularly with the scenes concerning Orton and Halliwell and the sets
(particularly of their flat with Halliwell\'s murals decorating every surface) are
terribly well done.'
```

In [13]:
```
df['sentiment'].value_counts()
```

Out[13]:
```
positive    5028
negative    4972
Name: sentiment, dtype: int64
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
review       0
sentiment    0
dtype: int64
```

In [15]:
```
df.duplicated().sum()
```

Out[15]:
```
17
```

In [16]:
```
df.drop_duplicates(inplace=True)
```

```
C:\Users\hi\AppData\Local\Temp\ipykernel_13568\3006716147.py:1: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.drop_duplicates(inplace=True)
```

In [18]:
```
df.duplicated().sum()
```

Out[18]:
```
0
```

In [19]:
```
import re
def remove_tags(raw_text):
    cleaned_text = re.sub(re.compile('<.*?>'), '', raw_text)
    return cleaned_text
```

In [20]:
```
df['review'] = df['review'].apply(remove_tags)
```

```
C:\Users\hi\AppData\Local\Temp\ipykernel_13568\2336150696.py:1: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['review'] = df['review'].apply(remove_tags)
```

In [21]:
```
df
```

Out[21]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The filming tec...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive
...	...	...
9995	Fun, entertaining movie about WWII German spy ...	positive
9996	Give me a break. How can anyone say that this ...	negative
9997	This movie is a bad movie. But after watching ...	negative
9998	This is a movie that was probably made to ente...	negative
9999	Smashing film about film-making. Shows the int...	positive

9983 rows × 2 columns

In [22]:
```
df['review'] = df['review'].apply(lambda x:x.lower())
```

```
C:\Users\hi\AppData\Local\Temp\ipykernel_13568\740760900.py:1: SettingWithCopyWarn
ing:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['review'] = df['review'].apply(lambda x:x.lower())
```

In [28]:
```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
sw_list = stopwords.words('english')

df['review'] = df['review'].apply(lambda x: [item for item in x.split() if item not
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\hi\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
C:\Users\hi\AppData\Local\Temp\ipykernel_13568\2672448455.py:6: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['review'] = df['review'].apply(lambda x: [item for item in x.split() if item
not in sw_list]).apply(lambda x:" ".join(x))
```

In [29]:
```
df
```

Out[29]:

	review	sentiment
0	one reviewers mentioned watching 1 oz episode ...	positive
1	wonderful little production. filming technique...	positive
2	thought wonderful way spend time hot summer we...	positive
3	basically there's family little boy (jake) thi...	negative
4	petter mattei's "love time money" visually stu...	positive
...	...	...
9995	fun, entertaining movie wwii german spy (julie...	positive
9996	give break. anyone say "good hockey movie"? kn...	negative
9997	movie bad movie. watching endless series bad h...	negative
9998	movie probably made entertain middle school, e...	negative
9999	smashing film film-making. shows intense stran...	positive

9983 rows × 2 columns

In [36]:
```
x = df.iloc[:,0:1]
y = df['sentiment']
```

In [37]:
```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()

y = encoder.fit_transform(y)
```

In [38]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=1)
```

In [39]:
```
from sklearn.feature_extraction.text import CountVectorizer
```

In [40]:
```
cv = CountVectorizer()
```

In [41]:
```
x_train_bow = cv.fit_transform(x_train['review']).toarray()
x_test_bow = cv.transform(x_test['review']).toarray()
```

In [42]:
```
x_train_bow.shape
```

Out[42]:
```
(7986, 48282)
```

In [43]:
```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()

gnb.fit(x_train_bow,y_train)
```

Out[43]:
```
GaussianNB()
```

In [44]:
```
y_pred = gnb.predict(x_test_bow)

from sklearn.metrics import accuracy_score,confusion_matrix
accuracy_score(y_test,y_pred)
```

Out[44]:
```
0.6324486730095142
```

In [45]:
```
confusion_matrix(y_test,y_pred)
```

Out[45]:
```
array([[717, 235],
       [499, 546]], dtype=int64)
```

In [46]:
```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()

rf.fit(x_train_bow,y_train)
```

Out[46]:
```
RandomForestClassifier()
```

In [47]:
```
y_pred = rf.predict(x_test_bow)
accuracy_score(y_test,y_pred)
```

Out[47]:
```
0.8412618928392589
```

In [48]:
```
cv = CountVectorizer(max_features=3000)

x_train_bow = cv.fit_transform(x_train['review']).toarray()
x_test_bow = cv.transform(x_test['review']).toarray()

rf = RandomForestClassifier()

rf.fit(x_train_bow,y_train)
y_pred = rf.predict(x_test_bow)
accuracy_score(y_test,y_pred)
```

Out[48]:
```
0.8347521281922884
```

In [49]:
```
cv = CountVectorizer(ngram_range=(1,2),max_features=5000)
```

```
x_train_bow = cv.fit_transform(x_train['review']).toarray()
x_test_bow = cv.transform(x_test['review']).toarray()

rf = RandomForestClassifier()

rf.fit(x_train_bow,y_train)
y_pred = rf.predict(x_test_bow)
accuracy_score(y_test,y_pred)
```

Out[49]:
```
0.8362543815723585
```