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

In [2]:
```
temp_df = pd.read_csv("D:\dowloads\IMDB Dataset.csv")
```

In [3]:
```
df = temp_df.iloc[:10000]
```

In [4]:
```
df.head()
```

Out[4]:

	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

In [5]:
```
df['review'][1]
```

Out[5]:
```
'A wonderful little production. <br /><br />The filming technique is very unassuming- very old-time-BBC fashion and gives a comforting, and sometimes discomfortin
g, sense of realism to the entire piece. <br /><br />The actors are extremely well chosen- Michael Sheen not only "has got all the polari" but he has all the voice
s down pat too! You can truly see the seamless editing guided by the references to Williams\' diary entries, not only is it well worth the watching but it is a ter
rificly written and performed piece. A masterful production about one of the great master\'s of comedy and his life. <br /><br />The realism really comes home with
the little things: the fantasy of the guard which, rather than use the traditional \'dream\' techniques remains solid then disappears. It plays on our knowledge an
d our senses, particularly with the scenes concerning Orton and Halliwell and the sets (particularly of their flat with Halliwell\'s murals decorating every surfac
e) are terribly well done.'
```

In [6]:
```
df['sentiment'].value_counts()
```

Out[6]:
```
positive    5028
negative    4972
Name: sentiment, dtype: int64
```

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
review       0
sentiment    0
dtype: int64
```

In [8]:
```
df.duplicated().sum()
```

Out[8]:
```
17
```

In [9]:
```
df.drop_duplicates(inplace=True)
```

```
C:\Users\Saket\AppData\Local\Temp\ipykernel_6188\3006716147.py:1: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df.drop_duplicates(inplace=True)
```

In [10]:
```
df.duplicated().sum()
```

Out[10]:
```
0
```

In [11]:
```
import re
def remove_tags(raw_text):
    cleaned_text = re.sub(re.compile('<.*?>'), '', raw_text)
    return cleaned_text
```

In [12]:
```
df['review'] = df['review'].apply(remove_tags)
```

```
C:\Users\Saket\AppData\Local\Temp\ipykernel_6188\2336150696.py:1: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.o
rg/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df['review'] = df['review'].apply(remove_tags)
```

In [13]:
```
df
```

Out[13]:

	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 [14]:
```
df['review'] = df['review'].apply(lambda x:x.lower())
```

```
C:\Users\Saket\AppData\Local\Temp\ipykernel_6188\740760900.py:1: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df['review'] = df['review'].apply(lambda x:x.lower())
```

In [15]:
```
from nltk.corpus import stopwords

sw_list = stopwords.words('english')

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

```
C:\Users\Saket\AppData\Local\Temp\ipykernel_6188\2826946130.py:5: SettingWithCopyWarning:
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/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/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 [16]:
```
df
```

Out[16]:

	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 [17]:
```
x = df.iloc[:,0:1]
y = df['sentiment']
```

In [18]:
```
from sklearn.preprocessing import LabelEncoder

encoder = LabelEncoder()

y = encoder.fit_transform(y)
```

In [19]:
```
x
```

Out[19]:

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

9983 rows × 1 columns

In [20]:
```
y
```

Out[20]:
```
array([1, 1, 1, ..., 0, 0, 1])
```

In [22]:
```
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 [23]:
```
X_train.shape
```

Out[23]:
```
(7986, 1)
```

In [24]:
```
# Applying BoW
from sklearn.feature_extraction.text import CountVectorizer
```

In [25]:
```
cv = CountVectorizer()
```

In [26]:
```
X_train_bow = cv.fit_transform(X_train['review']).toarray()
X_test_bow = cv.transform(X_test['review']).toarray()
```

In [27]:
```
X_train_bow.shape
```

Out[27]:
```
(7986, 48282)
```

In [28]:
```
from sklearn.metrics import accuracy_score,confusion_matrix
```

In [29]:
```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()

rf.fit(X_train_bow,y_train)
y_pred = rf.predict(X_test_bow)
accuracy_score(y_test,y_pred)
```

Out[29]:
```
0.8452679018527791
```

In [ ]:
```

```