In [2]:
```
import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import SnowballStemmer
import nltk
nltk.download('stopwords')
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB,MultinomialNB, BernoulliNB
from sklearn.metrics import accuracy_score
import pickle
```

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

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

```
(50000, 2)
```

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

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   review     50000 non-null  object
 1   sentiment  50000 non-null  object
dtypes: object(2)
memory usage: 781.4+ KB
```

In [5]:
```
#Check for no nulls
data.sentiment.value_counts()
```

Out[5]:
```
positive    25000
negative    25000
Name: sentiment, dtype: int64
```

In [6]:
```
# Label encode 0 for negitive and 1 for postive

data.sentiment.replace('positive', 1, inplace = True)
data.sentiment.replace('negative', 0, inplace = True)
```

In [7]:
```
# Check file
data.head(10)
```

Out[7]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	1
1	A wonderful little production. The...	1
2	I thought this was a wonderful way to spend ti...	1
3	Basically there's a family where a little boy ...	0
4	Petter Mattei's "Love in the Time of Money" is...	1
5	Probably my all-time favorite movie, a story o...	1
6	I sure would like to see a resurrection of a u...	1
7	This show was an amazing, fresh & innovative i...	0
8	Encouraged by the positive comments about this...	0
9	If you like original gut wrenching laughter yo...	1

In [8]:
```
data.review[1]
```

Out[8]:
```
'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 [9]:
```
# Pre Processing
# Removing HTML tags using regex rule

def clean(text):
    cleaned = re.compile(r'<.*?>')
    return re.sub(cleaned,'',text)


data.review = data.review.apply(clean)
data.review[1]
```

Out[9]:
```
'A wonderful little production. The filming technique is very unassuming- very old
-time-BBC fashion and gives a comforting, and sometimes discomforting, sense of re
alism to the entire piece. The actors are extremely well chosen- Michael Sheen not
only "has got all the polari" but he has all the voices down pat too! You can trul
y see the seamless editing guided by the references to Williams\' diary entries, n
ot only is it well worth the watching but it is a terrificly written and performed
piece. A masterful production about one of the great master\'s of comedy and his l
ife. The realism really comes home with the little things: the fantasy of the guar
d which, rather than use the traditional \'dream\' techniques remains solid then d
isappears. It plays on our knowledge and our senses, particularly with the scenes
concerning Orton and Halliwell and the sets (particularly of their flat with Halli
well\'s murals decorating every surface) are terribly well done.'
```

In [10]:
```
  # Remove special characters

def is_special(text):
    rem = ''
```

```
    for i in text:
        if i.isalnum():
            rem = rem + i
        else:
            rem = rem + ' '
    return rem

data.review = data.review.apply(is_special)
data.review[1]
```

Out[10]:
```
'A wonderful little production  The filming technique is very unassuming  very old
time BBC fashion and gives a comforting  and sometimes discomforting  sense of rea
lism to the entire piece  The actors are extremely well chosen  Michael Sheen not
only  has got all the polari  but he has all the voices down pat too  You can trul
y see the seamless editing guided by the references to Williams  diary entries  no
t only is it well worth the watching but it is a terrificly written and performed
piece  A masterful production about one of the great master s of comedy and his li
fe  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 disap
pears  It plays on our knowledge and our senses  particularly with the scenes conc
erning Orton and Halliwell and the sets  particularly of their flat with Halliwell
s murals decorating every surface  are terribly well done '
```

In [11]:
```
# convert every thing into lowercase
def is_lower(text):
    return text.lower()

data.review = data.review.apply(is_lower)
data.review[1]
```

Out[11]:
```
'a wonderful little production  the filming technique is very unassuming  very old
time bbc fashion and gives a comforting  and sometimes discomforting  sense of rea
lism to the entire piece  the actors are extremely well chosen  michael sheen not
only  has got all the polari  but he has all the voices down pat too  you can trul
y see the seamless editing guided by the references to williams  diary entries  no
t only is it well worth the watching but it is a terrificly written and performed
piece  a masterful production about one of the great master s of comedy and his li
fe  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 disap
pears  it plays on our knowledge and our senses  particularly with the scenes conc
erning orton and halliwell and the sets  particularly of their flat with halliwell
s murals decorating every surface  are terribly well done '
```

In [13]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Admin\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

Out[13]:
```
True
```

In [14]:
```
# Tokenize and remove stopwords
def rem_stopwords(text):
    stop_words = set(stopwords.words('english'))
    words = word_tokenize(str(text))
    return [w for w in words if w not in stop_words]

data.review = data.review.apply(rem_stopwords)
data.review[1]
```

Out[14]:
```
['wonderful',
 'little',
 'production',
 'filming',
 'technique',
 'unassuming',
 'old',
 'time',
 'bbc',
 'fashion',
 'gives',
 'comforting',
 'sometimes',
 'discomforting',
 'sense',
 'realism',
 'entire',
 'piece',
 'actors',
 'extremely',
 'well',
 'chosen',
 'michael',
 'sheen',
 'got',
 'polari',
 'voices',
 'pat',
 'truly',
 'see',
 'seamless',
 'editing',
 'guided',
 'references',
 'williams',
 'diary',
 'entries',
 'well',
 'worth',
 'watching',
 'terrificly',
 'written',
 'performed',
 'piece',
 'masterful',
 'production',
 'one',
 'great',
 'master',
 'comedy',
 'life',
 'realism',
 'really',
 'comes',
 'home',
 'little',
 'things',
 'fantasy',
 'guard',
 'rather',
 'use',
 'traditional',
 'dream',
 'techniques',
```

```
 'remains',
 'solid',
 'disappears',
 'plays',
 'knowledge',
 'senses',
 'particularly',
 'scenes',
 'concerning',
 'orton',
 'halliwell',
 'sets',
 'particularly',
 'flat',
 'halliwell',
 'murals',
 'decorating',
 'every',
 'surface',
 'terribly',
 'well',
 'done']
```

In [15]:
```
#Stemming

def stem_txt(text):
    ss = SnowballStemmer('english')
    return " ".join([ss.stem(w) for w in text])

data.review = data.review.apply(stem_txt)
data.review[1]
```

Out[15]:
```
'wonder littl product film techniqu unassum old time bbc fashion give comfort some
tim discomfort sens realism entir piec actor extrem well chosen michael sheen got
polari voic pat truli see seamless edit guid refer william diari entri well worth
watch terrif written perform piec master product one great master comedi life real
ism realli come home littl thing fantasi guard rather use tradit dream techniqu re
main solid disappear play knowledg sens particular scene concern orton halliwel se
t particular flat halliwel mural decor everi surfac terribl well done'
```

In [16]:
```
data.head()
# end of preprocessing of data
```

Out[16]:

	review	sentiment
0	one review mention watch 1 oz episod hook righ...	1
1	wonder littl product film techniqu unassum old...	1
2	thought wonder way spend time hot summer weeke...	1
3	basic famili littl boy jake think zombi closet...	0
4	petter mattei love time money visual stun film...	1

In [17]:
```
# ML Part
# creating the model
# creating bag of words (BOW)

x = np.array(data.iloc[:0].values)
y = np.array(data.sentiment.values)

cv = CountVectorizer(max_features = 1000)
```

```
x =cv.fit_transform(data.review).toarray()

print("X.shape = ", x.shape)
print("Y.shape = ", y.shape)

print(x)
```

```
X.shape =  (50000, 1000)
Y.shape =  (50000,)
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 1 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 1 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

In [18]:
```
# Split the data into training and testing

trainx, testx, trainy, testy = train_test_split(x,y,test_size =0.2, random_state =

print("Training data shapes: x = {}, y = {}".format(trainx.shape, trainy.shape))
print("Testing data shapes: x = {}, y = {}".format(testx.shape, testy.shape))
```

```
Training data shapes: x = (40000, 1000), y = (40000,)
Testing data shapes: x = (10000, 1000), y = (10000,)
```

In [19]:
```
# define models and train them

gnd, mnd, bnd = GaussianNB(), MultinomialNB(alpha = 1.0, fit_prior = True), Bernoul

gnd.fit(trainx, trainy)
mnd.fit(trainx, trainy)
bnd.fit(trainx, trainy)
```

Out[19]:
```
BernoulliNB()
```

In [20]:
```
# Run the test data and measure the accuracy

ypg =gnd.predict(testx)
ypm =mnd.predict(testx)
ypb =bnd.predict(testx)

print("Gaussian % = ", accuracy_score(testy,gnd.predict(testx))*100)
print("Multinomial % = ", accuracy_score(testy,ypm)*100)
print("Bernoulli % = ", accuracy_score(testy,ypb)*100)
```

```
Gaussian % =  78.43
Multinomial % =  83.1
Bernoulli % =  83.86
```

In [21]:
```
rev ="""Bharat has been described by its makers as one man’s life story unfolding p

Salman Khan plays the titular hero whom we follow from the age of eight until a lit

Bharat is directed by Ali Abbas Zafar, who chipped away at Salman’s larger-than-lif

f1 = clean(rev)
f2 = is_special(f1)
f3 = is_lower(f2)
f4 = rem_stopwords(f3)
f5 = stem_txt(f4)

bow, words = [], word_tokenize(f5)
```

```
for word in words:
    bow.append(words.count(word))

word_dict = cv.vocabulary_
pickle.dump(word_dict, open("bow.pkl",'wb'))

inp = []
for i in word_dict:
            inp.append(f5.count(i[0]))
y_pred = bnd.predict(np.array(inp).reshape(1,1000))
print("Bernoulli Prediction", y_pred)
```

```
Bernoulli Prediction [1]
```