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

In [48]:
```
#read data from csv file
data = pd.read_csv('../imdb_labelled.txt',sep ='\t',names=['comment','label'])
```

In [50]:
```
data.head(10)
```

Out[50]:

	comment	label
0	A very, very, very slow-moving, aimless movie ...	0
1	Not sure who was more lost - the flat characte...	0
2	Attempting artiness with black & white and cle...	0
3	Very little music or anything to speak of.	0
4	The best scene in the movie was when Gerardo i...	1
5	The rest of the movie lacks art, charm, meanin...	0
6	Wasted two hours.	0
7	Saw the movie today and thought it was a good ...	1
8	A bit predictable.	0
9	Loved the casting of Jimmy Buffet as the scien...	1

In [ ]:
```

```

In [ ]:
```

```

In [ ]:
```

```

In [51]:
```
#view data statistics using describe()
data.describe()
```

Out[51]:

	label
count	748.000000
mean	0.516043
std	0.500077
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

In [53]:
```
#view columns of the dataset
data.groupby('label').describe()
```

Out[53]:

	comment			
	count	unique	top	freq
label				
0	362	361	Not recommended.	2
1	386	384	Definitely worth checking out.	2

In [54]:
```
data['length']= data['comment'].apply(len)
```

In [56]:
```
#Count number of records
data.head()
```

Out[56]:

	comment	label	length
0	A very, very, very slow-moving, aimless movie ...	0	87
1	Not sure who was more lost - the flat characte...	0	99
2	Attempting artiness with black & white and cle...	0	188
3	Very little music or anything to speak of.	0	44
4	The best scene in the movie was when Gerardo i...	1	108

In [57]:
```
#view datatypes
type(data)
```

Out[57]:
```
pandas.core.frame.DataFrame
```

In [58]:
```
#select FDNY information boroughwise
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
```

In [62]:
```
def text_message(mess):
    no_pun= [char for char in mess if char not in string.punctuation]
    no_pun= ''.join(no_pun)
    return [word for word in no_pun.split() if word.lower() not in stopwords.words('english')]
```

In [63]:
```
import string
from  nltk.corpus import stopwords
```

In [66]:
```
bage_of_word = CountVectorizer(analyzer=text_message).fit(data['comment'])
```

In [67]:
```
comment_bage_of_word =bage_of_word.transform(data['comment'])
```

In [68]:
```
from sklearn.feature_extraction.text import TfidfTransformer
tfidfTransformer = TfidfTransformer().fit(comment_bage_of_word)
```

In [69]:
```
#view FDNY informationn for each borough
comment_ifidf= tfidfTransformer.transform(comment_bage_of_word)
print(comment_ifidf.shape)
```

```
(748, 3259)
```

In [71]:
```
from sklearn.naive_bayes import MultinomialNB
sentiment_detect_model= MultinomialNB().fit(comment_ifidf,data['label'])
```

In [74]:
```
comment =data['comment'][4]
bage_of_word_comment = bage_of_word.transform([comment])
tfidf =tfidfTransformer.transform(bage_of_word_comment)
```

In [75]:
```
print('predicted sentiment label',sentiment_detect_model.predict(tfidf)[0])
print('expected sentiment label',data.label[4])
```

```
predicted sentiment label 1
expected sentiment label 1
```

In [ ]:
```

```

In [ ]:
```

```