In [17]:

**import** nltk.classify.util

**from** nltk.classify **import** NaiveBayesClassifier

**from** nltk.corpus **import** movie\_reviews

*#nltk.download('movie\_reviews') downloaded the package for some functionality*

In [2]:

**def** extract\_features(word\_list):

**return** dict([(word, **True**) **for** word **in** word\_list])

In [7]:

**if** name **==**' main ':

*#Load positive and negative reviews*

positive\_fileids **=** movie\_reviews**.**fileids('pos') negative\_fileids **=** movie\_reviews**.**fileids('neg')

In [9]:

features\_positive **=** [(extract\_features(movie\_reviews**.**words(fileids**=**[f])),'Positive') **for** f **in** positive\_fileids] features\_negative **=** [(extract\_features(movie\_reviews**.**words(fileids**=**[f])),'Negative') **for** f **in** negative\_fileids]

In [10]:

*# Split the data into train and test (80/20)*

threshold\_factor **=** 0.8

threshold\_positive **=** int(threshold\_factor **\*** len(features\_positive)) threshold\_negative **=** int(threshold\_factor **\*** len(features\_negative))

In [11]:

features\_train **=** features\_positive[:threshold\_positive]**+**features\_negative[:threshold\_negative] features\_test **=** features\_positive[threshold\_positive:]**+**features\_negative[threshold\_negative:]

print("Number of training datapoints: ", len(features\_train)) print("Number of test datapoints: ", len(features\_test))

Number of training datapoints: 1600 Number of test datapoints: 400

In [12]:

classifier **=** NaiveBayesClassifier**.**train(features\_train)

print("Accuracy of the classifier: ", nltk**.**classify**.**util**.**accuracy(classifier, features\_test))

Accuracy of the classifier: 0.735

In [13]:

print("Top ten most informative words: ")

**for** item **in** classifier**.**most\_informative\_features()[:10]: print(item[0])

Top ten most informative words: outstanding

insulting

vulnerable ludicrous

uninvolving astounding avoids

fascination affecting

animators

In [14]:

*#Sample input reviews*

input\_reviews **=** [

"Started off as the greatest series of all time, but had the worst ending of all time.",

"Exquisite. 'Big Little Lies' takes us to an incredible journey with its emotional and intriguing storyline.", "I love Brooklyn 99 so much. It has the best crew ever!!",

"The Big Bang Theory and to me it's one of the best written sitcoms currently on network TV.",

"'Friends' is simply the best series ever aired. The acting is amazing.",

"SUITS is smart, sassy, clever, sophisticated, timely and immensely entertaining!",

"Cumberbatch is a fantastic choice for Sherlock Holmes-he is physically right (he fits the traditional reading of the charac "What sounds like a typical agent hunting serial killer, surprises with great characters, surprising turning points and amaz "This is one of the most magical things I have ever had the fortune of viewing.",

"I don't recommend watching this at all!"

]

In [15]:

print("Predictions: ")

**for** review **in** input\_reviews: print("\nReview:", review)

probdist **=** classifier**.**prob\_classify(extract\_features(review**.**split())) pred\_sentiment **=** probdist**.**max()

Review: Exquisite. 'Big Little Lies' takes us to an incredible journey with its emotional and intriguing storyline. Review: I love Brooklyn 99 so much. It has the best crew ever!!

Review: The Big Bang Theory and to me it's one of the best written sitcoms currently on network TV.

Review: 'Friends' is simply the best series ever aired. The acting is amazing.

Review: SUITS is smart, sassy, clever, sophisticated, timely and immensely entertaining!

Review: Cumberbatch is a fantastic choice for Sherlock Holmes-he is physically right (he fits the traditional reading of the cha racter) and he is a damn good actor

Review: What sounds like a typical agent hunting serial killer, surprises with great characters, surprising turning points and a mazing cast.This is one of the most magical things I have ever had the fortune of viewing.

Review: I don't recommend watching this at all!

In [16]:

print("Predictions: ")

**for** review **in** input\_reviews: print("\nReview:", review)

probdist **=** classifier**.**prob\_classify(extract\_features(review**.**split())) pred\_sentiment **=** probdist**.**max()

print("Predicted sentiment: ", pred\_sentiment)

print("Probability: ", round(probdist**.**prob(pred\_sentiment), 2))

Predicted sentiment: Negative Probability: 0.64

Review: Exquisite. 'Big Little Lies' takes us to an incredible journey with its emotional and intriguing storyline. Predicted sentiment: Positive

Probability: 0.89

Review: I love Brooklyn 99 so much. It has the best crew ever!! Predicted sentiment: Negative

Probability: 0.51

Review: The Big Bang Theory and to me it's one of the best written sitcoms currently on network TV. Predicted sentiment: Positive

Probability: 0.62

Review: 'Friends' is simply the best series ever aired. The acting is amazing. Predicted sentiment: Positive

Probability: 0.55

Review: SUITS is smart, sassy, clever, sophisticated, timely and immensely entertaining! Predicted sentiment: Positive

Probability: 0.82

Review: Cumberbatch is a fantastic choice for Sherlock Holmes-he is physically right (he fits the traditional reading of the cha racter) and he is a damn good actor

Predicted sentiment: Positive Probability: 1.0

Review: What sounds like a typical agent hunting serial killer, surprises with great characters, surprising turning points and a mazing cast.This is one of the most magical things I have ever had the fortune of viewing.

Predicted sentiment: Positive Probability: 0.95

Review: I don't recommend watching this at all! Predicted sentiment: Negative

Probability: 0.53

In [ ]: