In [1]:
```
import nltk
from nltk.corpus import stopwords
import csv
import numpy as np

stopwords = set(stopwords.words('english'))
```

In [2]:
```
import pandas as pd
df = pd.read_csv(r"C:\Users\raval\Downloads\full_training_dataset.csv",names=["sentiment","
```

In [3]:
```
positive_sentences = df[df['sentiment'] == 'positive']['sentence'].tolist()

# Create a list of tuples
positive_tuples = [(sentiment, sentence) for sentiment, sentence in zip(['positive'] * len(

print(positive_tuples)


# positive = df[df['sentiment'] == 'positive']['sentence'].tolist()
negative_sentences = df[df['sentiment'] == 'negative']['sentence'].tolist()

# Create a list of tuples
negative_tuples = [(sentiment, sentence) for sentiment, sentence in zip(['negative'] * len(

print(negative_tuples)
```

```
[('positive', 'the rock is destined to be the 21st century\'s new " conan " and that h
e\'s going to make a splash even greater than arnold schwarzenegger , jean-claud van d
amme or steven segal .'), ('positive', 'the gorgeously elaborate continuation of " the
lord of the rings " trilogy is so huge that a column of words cannot adequately descri
be co-writer/director peter jackson\'s expanded vision of j . r . r . tolkien\'s middl
e-earth .'), ('positive', 'effective but too-tepid biopic'), ('positive', 'if you some
times like to go to the movies to have fun , wasabi is a good place to start .'), ('po
sitive', "emerges as something rare , an issue movie that's so honest and keenly obser
ved that it doesn't feel like one ."), ('positive', 'the film provides some great insi
ght into the neurotic mindset of all comics -- even those who have reached the absolut
e top of the game .'), ('positive', 'offers that rare combination of entertainment and
education .'), ('positive', 'perhaps no picture ever made has more literally showed th
at the road to hell is paved with good intentions .'), ('positive', "steers turns in a
snappy screenplay that curls at the edges ; it's so clever you want to hate it . but h
e somehow pulls it off ."), ('positive', 'take care of my cat offers a refreshingly di
fferent slice of asian cinema .'), ('positive', 'this is a film well worth seeing , ta
lking and singing heads and all .'), ('positive', 'what really surprises about wisegir
ls is its low-key quality and genuine tenderness .'), ('positive', '( wendigo is ) why
we go to the cinema : to be fed through the eye , the heart , the mind .'), ('positiv
```

In [4]:
```
# Combine positive and negative tuples
Senti_tweets = []
for (sentiment, sentence) in positive_tuples + negative_tuples:
    words_filtered = [e.lower() for e in sentence.split() if len(e) >= 3]
    Senti_tweets.append((words_filtered, sentiment))

print(Senti_tweets)
```

```
[(['the', 'rock', 'destined', 'the', '21st', "century's", 'new', 'conan', 'and', 'tha
t', "he's", 'going', 'make', 'splash', 'even', 'greater', 'than', 'arnold', 'schwarzen
egger', 'jean-claud', 'van', 'damme', 'steven', 'segal'], 'positive'), (['the', 'gorge
ously', 'elaborate', 'continuation', 'the', 'lord', 'the', 'rings', 'trilogy', 'huge',
'that', 'column', 'words', 'cannot', 'adequately', 'describe', 'co-writer/director',
'peter', "jackson's", 'expanded', 'vision', "tolkien's", 'middle-earth'], 'positive'),
(['effective', 'but', 'too-tepid', 'biopic'], 'positive'), (['you', 'sometimes', 'lik
e', 'the', 'movies', 'have', 'fun', 'wasabi', 'good', 'place', 'start'], 'positive'),
(['emerges', 'something', 'rare', 'issue', 'movie', "that's", 'honest', 'and', 'keenl
y', 'observed', 'that', "doesn't", 'feel', 'like', 'one'], 'positive'), (['the', 'fil
m', 'provides', 'some', 'great', 'insight', 'into', 'the', 'neurotic', 'mindset', 'al
l', 'comics', 'even', 'those', 'who', 'have', 'reached', 'the', 'absolute', 'top', 'th
e', 'game'], 'positive'), (['offers', 'that', 'rare', 'combination', 'entertainment',
'and', 'education'], 'positive'), (['perhaps', 'picture', 'ever', 'made', 'has', 'mor
e', 'literally', 'showed', 'that', 'the', 'road', 'hell', 'paved', 'with', 'good', 'in
tentions'], 'positive'), (['steers', 'turns', 'snappy', 'screenplay', 'that', 'curls',
'the', 'edges', "it's", 'clever', 'you', 'want', 'hate', 'but', 'somehow', 'pulls', 'o
ff'], 'positive'), (['take', 'care', 'cat', 'offers', 'refreshingly', 'different', 'sl
ice', 'asian', 'cinema'], 'positive'), (['this', 'film', 'well', 'worth', 'seeing', 't
```

In [5]:
```
def get_words_in_tweets(tweets):
    all_words = []
    for (words, sentiment) in tweets:
        all_words.extend(words)
    return all_words
```

In [6]:
```
def get_word_features(wordlist):
    wordlist = nltk.FreqDist(wordlist)
    word_features = wordlist.keys()
    return word_features
```

In [7]:
```
# Assuming Senti_tweets is already defined
words_in_tweets = get_words_in_tweets(Senti_tweets)
word_features = get_word_features(words_in_tweets)
print(word_features)

word_features_filtered = [w for w in word_features if w not in stopwords]
print(word_features_filtered)
```

```
dict_keys(['the', 'rock', 'destined', '21st', "century's", 'new', 'conan', 'and', 'tha
t', "he's", 'going', 'make', 'splash', 'even', 'greater', 'than', 'arnold', 'schwarzen
egger', 'jean-claud', 'van', 'damme', 'steven', 'segal', 'gorgeously', 'elaborate', 'c
ontinuation', 'lord', 'rings', 'trilogy', 'huge', 'column', 'words', 'cannot', 'adequa
tely', 'describe', 'co-writer/director', 'peter', "jackson's", 'expanded', 'vision',
"tolkien's", 'middle-earth', 'effective', 'but', 'too-tepid', 'biopic', 'you', 'someti
mes', 'like', 'movies', 'have', 'fun', 'wasabi', 'good', 'place', 'start', 'emerges',
'something', 'rare', 'issue', 'movie', "that's", 'honest', 'keenly', 'observed', "does
n't", 'feel', 'one', 'film', 'provides', 'some', 'great', 'insight', 'into', 'neuroti
c', 'mindset', 'all', 'comics', 'those', 'who', 'reached', 'absolute', 'top', 'game',
'offers', 'combination', 'entertainment', 'education', 'perhaps', 'picture', 'ever',
'made', 'has', 'more', 'literally', 'showed', 'road', 'hell', 'paved', 'with', 'intent
ions', 'steers', 'turns', 'snappy', 'screenplay', 'curls', 'edges', "it's", 'clever',
'want', 'hate', 'somehow', 'pulls', 'off', 'take', 'care', 'cat', 'refreshingly', 'dif
ferent', 'slice', 'asian', 'cinema', 'this', 'well', 'worth', 'seeing', 'talking', 'si
nging', 'heads', 'what', 'really', 'surprises', 'about', 'wisegirls', 'its', 'low-ke
y', 'quality', 'genuine', 'tenderness', 'wendigo', 'why', 'fed', 'through', 'eye', 'he
art', 'mind', 'greatest', 'family-oriented', 'fantasy-adventure', 'ultimately', 'ponde
rs', 'reasons', 'need', 'stories', 'much', 'utterly', 'compelling', "'who", 'wrote',
```

In [8]:
```
def extract_features(document, word_features_filtered):
    document_words = set(document)
    features = {}
    for word in word_features_filtered:
        features['contains(%s)' % word] = (word in document_words)
    return features
```

In [9]:
```
# Assuming Senti_tweets and word_features_filtered are already defined
training_set = nltk.classify.apply_features(lambda doc: extract_features(doc, word_features
classifier = nltk.NaiveBayesClassifier.train(training_set)

test_tweet = 'This is a horrible book'
features = extract_features(test_tweet.split(), word_features_filtered)
sentiment = classifier.classify(features)
print("{}: Sentiment={}".format(test_tweet, sentiment))
```

```
This is a horrible book: Sentiment=negative
```

In [21]:
```
test_tweet = 'explaination is very good'
features = extract_features(test_tweet.split(), word_features_filtered)
sentiment = classifier.classify(features)
print("{}: Sentiment={}".format(test_tweet, sentiment))
```

```
explaination is very good: Sentiment=positive
```

In [ ]: