Lab Assignment - 4

In [ ]:
```
import nltk
from nltk.corpus import stopwords
stopwords=set(stopwords.words('english'))

pos_tweets=[('It is not impossible', 'positive'),
                    ('You are my lovely friend', 'Positive'),
                    ('She is beautiful girl', 'Positive'),
                    ('He is looking handsome', 'Positive'),
                    ('Exercise is good for health', 'Positive'),
                    ('Today\'s weather is fantastic', 'Positive'),
                    ('I love Mango', 'Positive')]

neg_tweets=[('You are my enemy friend', 'Negative'),
                    ('She is looking ugly ', 'Negative'),
                    ('He is looking horrible', 'Negative'),
                    ('Sleeping more makes you lazy', 'Negative'),
                    ('Today\'s weather is very bad', 'Negative'),
                    ('I hate Banana', 'Negative')]
#print(pos_tweets)
#print(neg_tweets)
```

In [ ]:
```
Senti_tweets=[]
for (words, sentiment) in pos_tweets + neg_tweets:
    words_filtered=[e.lower() for e in words.split() if len(e)>=3]
    Senti_tweets.append((words_filtered, sentiment))
print(Senti_tweets)

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


def get_word_features(wordlist):
    wordlist=nltk.FreqDist(wordlist)
    word_features=wordlist.keys()
    return word_features
```

In [ ]:
```
word_features=get_word_features(get_words_in_tweets(Senti_tweets))
print(word_features)

word_features_filtered=[]
for w in word_features:
    if w not in stopwords:
        word_features_filtered.append(w)

print(word_features_filtered)
```

In [4]:
```
def extract_features(document):
    document_words=set(document)
    features={}
    for word in word_features_filtered:
        features['contains(%s)' %word] = (word in document_words)
    return features

training_set = nltk.classify.apply_features(extract_features, Senti_tweets)
classifier = nltk.NaiveBayesClassifier.train(training_set)

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

```
[(['not', 'impossible'], 'positive'), (['you', 'are', 'lovely', 'friend'], 'Positive'),
(['she', 'beautiful', 'girl'], 'Positive'), (['looking', 'handsome'], 'Positive'), (['exer
cise', 'good', 'for', 'health'], 'Positive'), (["today's", 'weather', 'fantastic'], 'Posit
ive'), (['love', 'mango'], 'Positive'), (['you', 'are', 'enemy', 'friend'], 'Negative'),
(['she', 'looking', 'ugly'], 'Negative'), (['looking', 'horrible'], 'Negative'), (['sleepi
ng', 'more', 'makes', 'you', 'lazy'], 'Negative'), (["today's", 'weather', 'very', 'bad'],
'Negative'), (['hate', 'banana'], 'Negative')]
dict_keys(['not', 'impossible', 'you', 'are', 'lovely', 'friend', 'she', 'beautiful', 'gir
l', 'looking', 'handsome', 'exercise', 'good', 'for', 'health', "today's", 'weather', 'fan
tastic', 'love', 'mango', 'enemy', 'ugly', 'horrible', 'sleeping', 'more', 'makes', 'laz
y', 'very', 'bad', 'hate', 'banana'])
['impossible', 'lovely', 'friend', 'beautiful', 'girl', 'looking', 'handsome', 'exercise',
'good', 'health', "today's", 'weather', 'fantastic', 'love', 'mango', 'enemy', 'ugly', 'ho
rrible', 'sleeping', 'makes', 'lazy', 'bad', 'hate', 'banana']
This is a horrible book: Sentiment=Negative
```

In [2]:
```

```

```
'nltk.download' is not recognized as an internal or external command,
operable program or batch file.
```

In [3]:
```
>>> import nltk
>>> nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\raval\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Out[3]:
```
True
```

```
In [13]: import csv

         # Read the CSV file
         csv_file_path = r"C:\Users\raval\Downloads\full_training_dataset.csv"

         # Assuming your CSV file has columns named 'Sentence' and 'Label'
         sentences = []
         labels = []

         with open(csv_file_path, 'r',encoding='latin-1') as csv_file:
             csv_reader = csv.reader(csv_file)
             next(csv_reader)  # Skip the header row if it exists
             for row in csv_reader:
                 sentence = row[0]
                 label = row[1]
                 sentences.append(sentence)
                 labels.append(label)

         # Tokenize and filter words
         tokenized_sentences = [([e.lower() for e in sentence.split() if len(e) >= 3], label) for se

         # Apply the trained classifier
         results = []

         for sentence, label in tokenized_sentences:
             test_features = extract_features(sentence)
             sentiment = classifier.classify(test_features)
             results.append((sentence, sentiment))

         # Print the results
         for sentence, sentiment in results:
             print("{}: Sentiment={}".format(sentence, sentiment))
```

```
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
['positive']: Sentiment=Positive
```

In [18]:
```python
import nltk
from nltk.corpus import stopwords
import csv

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

# Your positive and negative tweets
pos_tweets = [('It is not impossible', 'positive'),
              ('You are my lovely friend', 'positive'),
              ('She is a beautiful girl', 'positive'),
              ('He is looking handsome', 'positive'),
              ('Exercise is good for health', 'positive'),
              ('Today\'s weather is fantastic', 'positive'),
              ('I love Mango', 'positive')]

neg_tweets = [('You are my enemy friend', 'negative'),
              ('She is looking ugly', 'negative'),
              ('He is looking horrible', 'negative'),
              ('Sleeping more makes you lazy', 'negative'),
              ('Today\'s weather is very bad', 'negative'),
              ('I hate Banana', 'negative')]

# Combine positive and negative tweets
Senti_tweets = []
for (words, sentiment) in pos_tweets + neg_tweets:
    words_filtered = [e.lower() for e in words.split() if len(e) >= 3]
    Senti_tweets.append((words_filtered, sentiment))

# Define functions
def get_words_in_tweets(tweets):
    all_words = []
    for (words, sentiment) in tweets:
        all_words.extend(words)
    return all_words

def get_word_features(wordlist):
    wordlist = nltk.FreqDist(wordlist)
    word_features = wordlist.keys()
    return word_features

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

# Get word features
word_features = get_word_features(get_words_in_tweets(Senti_tweets))
word_features_filtered = [w for w in word_features if w not in stopwords]

# Train the classifier
training_set = nltk.classify.apply_features(lambda doc: extract_features(doc, word_features
classifier = nltk.NaiveBayesClassifier.train(training_set)

# Apply the classifier to new data from the CSV file
csv_file_path = r"C:\Users\raval\Downloads\full_training_dataset.csv"

with open(csv_file_path, 'r', encoding='latin-1') as csv_file:
    csv_reader = csv.reader(csv_file)
    next(csv_reader)  # Skip the header row if it exists
    for row in csv_reader:
        test_tweet = row[0]
        features = extract_features(test_tweet.split(), word_features_filtered)
        sentiment = classifier.classify(features)
        print(f"{test_tweet}: Sentiment={sentiment.capitalize()}")
```

```
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
positive: Sentiment=Positive
```

In [25]:
```python
import nltk
from nltk.corpus import stopwords
import csv
import numpy as np

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

In [23]:
```python
import pandas as pd

df = pd.read_csv(r"C:\Users\raval\Downloads\full_training_dataset.csv",names=["sentiment","
```

In [24]:
```python
df
```

Out[24]:

	sentiment	sentence
0	positive	the rock is destined to be the 21st century's ...
1	positive	the gorgeously elaborate continuation of " the...
2	positive	effective but too-tepid biopic
3	positive	if you sometimes like to go to the movies to h...
4	positive	emerges as something rare , an issue movie tha...
...	...	...
21599	neutral	@madtruckman 'Modern Day Autograph", I like th...
21600	neutral	62 Ways to Use #Twitter for Business: http://t...
21601	neutral	Log off #Facebook On #Twitter , But I Think i'...
21602	neutral	"#twitter's dumb, I don't like it." Hush up, ...
21603	neutral	It's almost 4:20. Where is your bong? Is it pa...

21604 rows × 2 columns

In [26]:
```
df[df.sentiment=="positive"]
```

Out[26]:

	sentiment	sentence
0	positive	the rock is destined to be the 21st century's ...
1	positive	the gorgeously elaborate continuation of " the...
2	positive	effective but too-tepid biopic
3	positive	if you sometimes like to go to the movies to h...
4	positive	emerges as something rare , an issue movie tha...
...	...	...
19328	positive	goodmorning, preparing for conference call wit...
19329	positive	she makes every everything bad in my life seem...
19330	positive	&quot;We'll be a beets cover band&quot;. I wou...
19331	positive	Yep, I'm receiving DMs, so at least some of yo...
19332	positive	Hi Rhian this is my first post on twitter so h...

9667 rows × 2 columns

In [27]:
```
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', '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
e', 'one of the greatest family-oriented , fantasy-adventure movies ever .'), ('positi
```

In [29]:
```python
# 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)
```

```
[('negative', 'simplistic , silly and tedious .'), ('negative', "it's so laddish and j
uvenile , only teenage boys could possibly find it funny ."), ('negative', 'exploitati
ve and largely devoid of the depth or sophistication that would make watching such a g
raphic treatment of the crimes bearable .'), ('negative', '[garbus] discards the poten
tial for pathological study , exhuming instead , the skewed melodrama of the circumsta
ntial situation .'), ('negative', 'a visually flashy but narratively opaque and emotio
nally vapid exercise in style and mystification .'), ('negative', "the story is also a
s unoriginal as they come , already having been recycled more times than i'd care to c
ount ."), ('negative', "about the only thing to give the movie points for is bravado -
- to take an entirely stale concept and push it through the audience's meat grinder on
e more time ."), ('negative', 'not so much farcical as sour .'), ('negative', 'unfortu
nately the story and the actors are served with a hack script .'), ('negative', 'all t
he more disquieting for its relatively gore-free allusions to the serial murders , but
 it falls down in its attempts to humanize its subject .'), ('negative', 'a sentimental
mess that never rings true .'), ('negative', 'while the performances are often engagin
g , this loose collection of largely improvised numbers would probably have worked bet
ter as a one-hour tv documentary .'), ('negative', 'interesting , but not compelling 
.'), ('negative', 'on a cutting room floor somewhere lies . . . footage that might hav
e made no such thing a trenchant , ironic cultural satire instead of a frustrating mis
```

In [32]:
```python
# positive = df[df['sentiment'] == 'positive']['sentence'].tolist()
neutral_sentences = df[df['sentiment'] == 'neutral']['sentence'].tolist()

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

print(neutral_tuples)
```

```
[('neutral', '@Late_Show I would have watched but the folks at @apple have a jihad aga
inst adobe flash. Plse consider a YouTube link in future on UR site'), ('neutral', 'RT
@rdingwell: .@Apple has a record quarter and because a bunch of professional guessers
(aka analysts) wanted more, its a disappointmen ...'), ('neutral', "Hey @apple, androi
ds releasing brand new state of the art phones, whens your new phone come out? What's
that? (cont) http://t.co/2sko9l3d"), (http://t.co/2sko9l3d"),) ('neutral', '.@Apple ha
s a record quarter and because a bunch of professional guessers (aka analysts) wanted
more, its a disappointment #wtf'), ('neutral', "@Apple how fun wouldn't it be if it wa
s possible to integrate  ( soon to be named ) with notifications?"), ('neutral', 'Inte
resting read on war b/w @Apple & @Samsung- http://t.co/Vt9d24Yi (http://t.co/Vt9d24Yi)
-using latter, agree lack of innovation, but better specs at same price!'), ('neutra
l', 'RT @adamnash: The takeaway from the @Apple earnings call?  Even Apple needs a new
iPhone release every 12 months to stay competitive. cc ...'), ('neutral', 'The takeawa
y from the @Apple earnings call?  Even Apple needs a new iPhone release every 12 month
s to stay competitive. cc: @hblodget'), ('neutral', "Today's headline: @apple reports
lower 4Q earnings. Headline in 3 months: @Apple reports record Q1 earnings."), ('neutr
al', 'Win an @Apple iPod Touch from @Mommy_gaga, get the @Pampers Hello World Baby Mem
ories App! http://t.co/XVcch60s (http://t.co/XVcch60s) #PampersHelloApps"'), ('neutra
l', '@apple expanded the app store to more than 20 new countries in the december quart
```

```
In [37]: # Combine positive and negative tweets
         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))
         Senti_tweets
```

```
Out[37]: [(['the',
            'rock',
            'destined',
            'the',
            '21st',
            "century's",
            'new',
            'conan',
            'and',
            'that',
            "he's",
            'going',
            'make',
            'splash',
            'even',
            'greater',
            'than',
            'arnold',
            'schwarzenegger',
```

```
In [38]: def get_words_in_tweets(tweets):
             all_words=[]
             for (sentiment, sentence) in Senti_tweets:
                 all_words.extend(words)
             return (all_words)

         def get_word_features(wordlist):
             wordlist=nltk.FreqDist(wordlist)
             word_features=wordlist.keys()
             return word_features

         word_features=get_word_features(get_words_in_tweets(Senti_tweets))
         print(word_features)

         word_features_filtered=[]
         for w in word_features:
             if w not in stopwords:
                 word_features_filtered.append(w)

         print(word_features_filtered)
```

```
dict_keys(['I', ' ', 'h', 'a', 't', 'e', 'B', 'n'])
['I', ' ', 'h', 'e', 'B', 'n']
```

```
In [ ]:
```

```
In [ ]:
```

```
In [39]: def extract_features(document):
             document_words=set(document)
             features={}
             for word in word_features_filtered:
                 features['contains(%s)' %word] = (word in document_words)
             return features

         training_set = nltk.classify.apply_features(extract_features, Senti_tweets)
         classifier = nltk.NaiveBayesClassifier.train(training_set)

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

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

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```