In [1]:

```
pip install nltk==3.3
```

```
Collecting nltk==3.3
  Downloading nltk-3.3.0.zip (1.4 MB)
Requirement already satisfied: six in c:\users\nb291\anaconda3\lib\site-packages (from nl
tk==3.3) (1.14.0)
Building wheels for collected packages: nltk
  Building wheel for nltk (setup.py): started
  Building wheel for nltk (setup.py): finished with status 'done'
  Created wheel for nltk: filename=nltk-3.3-py3-none-any.whl size=1394475 sha256=dd5eb9d1
5ef660376a9fcf3fe0717e85209a697dc63bd0d86e0226c78c51392d
  Stored in directory: c:\users\nb291\appdata\local\pip\cache\wheels\9b\fd\0c\d92302c876e
5de87ebd7fc0979d82edb93e2d8d768bf71fac4
Successfully built nltk
Installing collected packages: nltk
  Attempting uninstall: nltk
    Found existing installation: nltk 3.4.5
    Uninstalling nltk-3.4.5:
      Successfully uninstalled nltk-3.4.5
Successfully installed nltk-3.3
Note: you may need to restart the kernel to use updated packages.
```

In [2]:

```
import nltk
```

In [3]:

```
nltk.download('twitter_samples')
```

```
[nltk_data] Downloading package twitter_samples to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\twitter_samples.zip.
```

Out[3]:

```
True
```

In [4]:

```
nltk.download('twitter_samples')
```

```
[nltk_data] Downloading package twitter_samples to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Package twitter_samples is already up-to-date!
```

Out[4]:

```
True
```

In [6]:

```
from nltk.corpus import twitter_samples
```

In [7]:

```
positive_tweets = twitter_samples.strings('positive_tweets.json')
negative_tweets = twitter_samples.strings('negative_tweets.json')
text = twitter_samples.strings('tweets.20150430-223406.json')
```

In [8]:

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

```
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

Out[8]:

```
True
```

In [9]:

```
tweet_tokens = twitter_samples.tokenized('positive_tweets.json')
```

In [10]:

```
[0]

print(tweet_tokens[0])
```

```
['#FollowFriday', '@France_Inte', '@PKuchly57', '@Milipol_Paris', 'for', 'being', 'top',
'engaged', 'members', 'in', 'my', 'community', 'this', 'week', ':)']
```

In [12]:

```
import nltk
nltk.download('wordnet')
nltk.download('averaged_perceptron_tagger')        #normalising the data
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\wordnet.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
```

Out[12]:

```
True
```

In [13]:

```
from nltk.tag import pos_tag
from nltk.corpus import twitter_samples

tweet_tokens = twitter_samples.tokenized('positive_tweets.json')
print(pos_tag(tweet_tokens[0]))
```

```
[('#FollowFriday', 'JJ'), ('@France_Inte', 'NNP'), ('@PKuchly57', 'NNP'), ('@Milipol_Pari
s', 'NNP'), ('for', 'IN'), ('being', 'VBG'), ('top', 'JJ'), ('engaged', 'VBN'), ('members
', 'NNS'), ('in', 'IN'), ('my', 'PRP$'), ('community', 'NN'), ('this', 'DT'), ('week', 'N
N'), (':)', 'NN')]
```

In [14]:

```
#NNP: Noun, proper, singular
#NN: Noun, common, singular or mass
#IN: Preposition or conjunction, subordinating
#VBG: Verb, gerund or present participle
#VBN: Verb, past participle
```

In [15]:

```
...

from nltk.tag import pos_tag
from nltk.stem.wordnet import WordNetLemmatizer

def lemmatize_sentence(tokens):
    lemmatizer = WordNetLemmatizer()
    lemmatized_sentence = []
    for word, tag in pos_tag(tokens):
        if tag.startswith('NN'):
            pos = 'n'
        elif tag.startswith('VB'):
            pos = 'v'
        else:
            pos = 'a'
        lemmatized_sentence.append(lemmatizer.lemmatize(word, pos))
    return lemmatized_sentence

print(lemmatize_sentence(tweet_tokens[0]))
```

```
['#FollowFriday', '@France_Inte', '@PKuchly57', '@Milipol_Paris', 'for', 'be', 'top', 'en
gage', 'member', 'in', 'my', 'community', 'this', 'week', ':)']
```

In [16]:

```
#removing noise
```

In [17]:

```
...

import re, string

def remove_noise(tweet_tokens, stop_words = ()):

    cleaned_tokens = []

    for token, tag in pos_tag(tweet_tokens):
        token = re.sub('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+#]|[!*\(\),]|'\
                       '(?:%[0-9a-fA-F][0-9a-fA-F]))+','', token)
        token = re.sub("(@[A-Za-z0-9_]+)","", token)

        if tag.startswith("NN"):
            pos = 'n'
        elif tag.startswith('VB'):
            pos = 'v'
        else:
            pos = 'a'

        lemmatizer = WordNetLemmatizer()
        token = lemmatizer.lemmatize(token, pos)

        if len(token) > 0 and token not in string.punctuation and token.lower() not in s
top_words:
            cleaned_tokens.append(token.lower())
    return cleaned_tokens
```

In [18]:

```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\nb291\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

Out[18]:

```
True
```

In [19]:

```
...
from nltk.corpus import stopwords
stop_words = stopwords.words('english')

print(remove_noise(tweet_tokens[0], stop_words))
```

```
['#followfriday', 'top', 'engage', 'member', 'community', 'week', ':)']
```

In [20]:

```
from nltk.corpus import stopwords
stop_words = stopwords.words('english')

#print(remove_noise(tweet_tokens[0], stop_words))
```

```
positive_tweet_tokens = twitter_samples.tokenized('positive_tweets.json')
negative_tweet_tokens = twitter_samples.tokenized('negative_tweets.json')

positive_cleaned_tokens_list = []
negative_cleaned_tokens_list = []

for tokens in positive_tweet_tokens:
    positive_cleaned_tokens_list.append(remove_noise(tokens, stop_words))

for tokens in negative_tweet_tokens:
    negative_cleaned_tokens_list.append(remove_noise(tokens, stop_words))
```

In [21]:

```
...
print(positive_tweet_tokens[500])
print(positive_cleaned_tokens_list[500])
```

```
['Dang', 'that', 'is', 'some', 'rad', '@AbzuGame', '#fanart', '!', ':D', 'https://t.co/bI
8k8tb9ht']
['dang', 'rad', '#fanart', ':d']
```

In [22]:

```
#Determining Word Density



...

def get_all_words(cleaned_tokens_list):
    for tokens in cleaned_tokens_list:
        for token in tokens:
            yield token

all_pos_words = get_all_words(positive_cleaned_tokens_list)
```

In [23]:

```
from nltk import FreqDist

freq_dist_pos = FreqDist(all_pos_words)
print(freq_dist_pos.most_common(10))
```

```
[(':)', 3691), (':-)', 701), (':d', 658), ('thanks', 388), ('follow', 357), ('love', 333)
, ('...', 290), ('good', 283), ('get', 263), ('thank', 253)]
```

In [24]:

```
#Preparing Data for the Model
```

In [25]:

```
...
def get_tweets_for_model(cleaned_tokens_list):
    for tweet_tokens in cleaned_tokens_list:
        yield dict([token, True] for token in tweet_tokens)

positive_tokens_for_model = get_tweets_for_model(positive_cleaned_tokens_list)
negative_tokens_for_model = get_tweets_for_model(negative_cleaned_tokens_list)
```

In [26]:

```
#Splitting the Dataset for Training and Testing the Model
```

In [27]:

```
...
import random

positive_dataset = [(tweet_dict, "Positive")
                     for tweet_dict in positive_tokens_for_model]
```

```
negative_dataset = [(tweet_dict, "Negative")
                     for tweet_dict in negative_tokens_for_model]

dataset = positive_dataset + negative_dataset

random.shuffle(dataset)

train_data = dataset[:7000]
test_data = dataset[7000:]
```

In [28]:

```
#Building and Testing the Model
```

In [29]:

```
...
from nltk import classify
from nltk import NaiveBayesClassifier
classifier = NaiveBayesClassifier.train(train_data)

print("Accuracy is:", classify.accuracy(classifier, test_data))

print(classifier.show_most_informative_features(10))
```

```
Accuracy is: 0.997
Most Informative Features
                      :( = True           Negati : Positi =   2056.6 : 1.0
                      :) = True           Positi : Negati =   1663.4 : 1.0
                     sad = True           Negati : Positi =     24.3 : 1.0
                follower = True           Positi : Negati =     23.1 : 1.0
                     bam = True           Positi : Negati =     16.4 : 1.0
                  arrive = True           Positi : Negati =     15.6 : 1.0
                     x15 = True           Negati : Positi =     15.6 : 1.0
                followed = True           Negati : Positi =     14.6 : 1.0
               community = True           Positi : Negati =     14.4 : 1.0
                    blog = True           Positi : Negati =     13.0 : 1.0
None
```

In [30]:

```
#check how the model performs on random tweets from Twitter
```

In [31]:

```
...
from nltk.tokenize import word_tokenize

custom_tweet = "I ordered just once from TerribleCo, they screwed up, never used the app
again."

custom_tokens = remove_noise(word_tokenize(custom_tweet))

print(classifier.classify(dict([token, True] for token in custom_tokens)))
```

```
Negative
```