AIT 580- NLP Classification
Siva Swetha Yalamanchili
G01057485

---

nltk package installation:

In [2]:
```
import nltk
```

In [3]:
```
def format_sentence(sent):
    return ({word: True for word in nltk.word_tokenize(sent)})
```

Sample sentence:

In [5]:
```
print(format_sentence("How you doin?"))
```

```
{'How': True, 'you': True, 'doin': True, '?': True}
```

In [7]:
```
pos = []
with open ("pos_tweets.txt",encoding='utf8') as f:
    for i in f:
        pos.append([format_sentence(i),'pos'])

neg = []
with open ("neg_tweets.txt",encoding='utf8') as f:
    for i in f:
        neg.append([format_sentence(i),'neg'])
```

In [9]:
```
pos = []
with open("/users/swetha.y/documents/pos_tweets.txt",encoding='utf8') as f:
    for i in f:
        pos.append([format_sentence(i), 'pos'])

neg = []
with open("/users/swetha.y/documents/neg_tweets.txt",encoding='utf8') as f:
    for i in f:
        neg.append([format_sentence(i), 'neg'])
```

In [11]:
```
training = pos[:int((.8)*len(pos))] + neg[:int((.8)*len(neg))]
test = pos[int((.8)*len(pos)):] + neg[int((.8)*len(neg)):]
```

In [13]:
```
from nltk.classify import NaiveBayesClassifier
classifier = NaiveBayesClassifier.train(training)
```

In [15]:
```
classifier.show_most_informative_features()
```

```
Most Informative Features
                      no = True              neg : pos    =     19.4 : 1.0
                    love = True              pos : neg    =     19.0 : 1.0
                 awesome = True              pos : neg    =     17.2 : 1.0
                headache = True              neg : pos    =     16.2 : 1.0
                      Hi = True              pos : neg    =     12.7 : 1.0
                   Thank = True              pos : neg    =      9.7 : 1.0
                     New = True              pos : neg    =      9.7 : 1.0
                     fan = True              pos : neg    =      9.7 : 1.0
               beautiful = True              pos : neg    =      9.7 : 1.0
                    haha = True              pos : neg    =      9.3 : 1.0
```

Tweets:

```
In [19]: Tweet1 = "@Lakers ready to win tonight!!!"
         print(classifier.classify(format_sentence(Tweet1)))
pos

In [21]: Tweet2 = "@alexbigman you left without saying hi!"
         print(classifier.classify(format_sentence(Tweet2)))
neg

In [23]: Tweet3 = "@ashleyskyy but I wanted a margarita too!"
         print(classifier.classify(format_sentence(Tweet3)))
neg

In [25]: Tweet4 = "@oprah, nite! that's a really cute pup by the way"
         print(classifier.classify(format_sentence(Tweet4)))
pos

In [27]: Tweet5 = "I love Christmas parties"
         print(classifier.classify(format_sentence(Tweet5)))
pos

In [29]: Tweet6 = "@Chrxs sick sick sick  No more McDonalds ever again. SUBWAY!"
         print(classifier.classify(format_sentence(Tweet6)))
neg

In [36]: Tweet7 = "@Cibaby sorry hear bout the cavs"
         print(classifier.classify(format_sentence(Tweet7)))
neg

In [37]: Tweet8 = "@teleken It's a feat of USB engineering! Makes every day a party"
         print(classifier.classify(format_sentence(Tweet8)))
pos

In [38]: Tweet9 = "Twin cities . . Can't wait to see ya later tonight"
         print(classifier.classify(format_sentence(Tweet9)))
pos

In [43]: Tweet10 = "it's never too late to go back to bed"
         print(classifier.classify(format_sentence(Tweet10)))
neg

In [45]: from nltk.classify.util import accuracy
         print(accuracy(classifier, test))
0.8308457711442786
```

Observations:

- The accuracy obtained by the set of tweets tested is 0.8308457711442786 which is 83%.
- Tweet 10 is a positive tweet but, it was misinterpreted due to the word 'never'.
- The accuracy may be improved if the interpretation is not solely done using the keywords but by the meaning of the sentence.

Ambiguous News Headlines:

```
In [47]: AH1 = "supreme court allows travel ban's full enforcement as challenges continue"
         print(classifier.classify(format_sentence(AH1)))
```

```
neg
```

```
In [49]: AH2 = "India won the test series with sri lanka"
         print(classifier.classify(format_sentence(AH2)))
```

```
pos
```

```
In [51]: AH3 = "Noah stays on as France Davis cup skipper"
         print(classifier.classify(format_sentence(AH3)))
```

```
pos
```

```
In [52]: from nltk.classify.util import accuracy
         print(accuracy(classifier, test))
```

```
0.8308457711442786
```

Observations:

- The accuracy for the ambiguous news headlines is still 83%.
- Here one of the three headlines got a negative output.
- The accuracy of the model can be improved by applying more training.