Name:Rahul Gudivada Reg no: 19BCE2469

TASK 1-LEXICONS

In [1]:

```
#Lexicon->collection of words/phrases+information
#Lexicon has lexical entires->each entry is word/phase
import nltk
#1. Stopwords
from nltk.corpus import stopwords
stopwords.words('english')
```

Out[1]:

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
```

In [2]:

```
stopwords.words('spanish')
```

Out[2]:

```
['de',
 'la',
 'que',
 'el',
 'en',
 'y',
 'a',
 'los',
 'del',
 'se',
 'las',
 'por',
 'un',
 'para',
 'con',
 'no',
 'una',
 'su',
```

In [3]:

```
stopwords.words('french')
```

Out[3]:

```
['au',
 'aux',
 'avec',
 'ce',
 'ces',
 'dans',
 'de',
 'des',
 'du',
 'elle',
 'en',
 'et',
 'eux',
 'il',
 'ils',
 'je',
 'la',
 'le',
```

In [4]:

```
#2. CMU Wordlist
import nltk
entries=nltk.corpus.cmudict.entries()
len(entries)
```

Out[4]:

```
133737
```

In [5]:

```
print(entries)
```

```
[('a', ['AH0']), ('a.', ['EY1']), ('a', ['EY1']), ...]
```

In [ ]:

In [6]:

```
#3. Wordnet
from nltk.corpus import wordnet as wn
wn.synsets('dog') #good
```

Out[6]:

```
[Synset('dog.n.01'),
 Synset('frump.n.01'),
 Synset('dog.n.03'),
 Synset('cad.n.01'),
 Synset('frank.n.02'),
 Synset('pawl.n.01'),
 Synset('andiron.n.01'),
 Synset('chase.v.01')]
```

In [7]:

```
wn.synset('frank.n.02').lemma_names()
```

Out[7]:

```
['frank',
 'frankfurter',
 'hotdog',
 'hot_dog',
 'dog',
 'wiener',
 'wienerwurst',
 'weenie']
```

Task 2- Simple Text Classifier

In [8]:

```
def gender_features(word):
    return {'last_letter':word[-1]}
```

In [9]:

```
gender_features('Obama')
```

Out[9]:

```
{'last_letter': 'a'}
```

In [10]:

```
from nltk.corpus import names
labeled_names=([(name,'male') for name in names.words('male.txt')]+[(name,'female') for nam
```

In [11]:

```
import random
random.shuffle(labeled_names)
```

In [12]:

```
featuresets=[(gender_features(n),gender) for (n,gender) in labeled_names]
```

In [13]:

```
train_set,test_test=featuresets[500:],featuresets[:500]
```

In [14]:

```
import nltk
classifier=nltk.NaiveBayesClassifier.train(train_set)
```

In [15]:

```
classifier.classify(gender_features('Rahul'))
```

Out[15]:

```
'male'
```

In [16]:

```
print(nltk.classify.accuracy(classifier,test_test))
```

```
0.728
```

Task 3-Vectorizer

In [17]:

```
from sklearn.feature_extraction.text import CountVectorizer
```

In [18]:

```
vect=CountVectorizer(binary=True)
corpus=["Tessaract is good optical character recognition engine", "optical character recogn
vect.fit(corpus)
```

Out[18]:

```
CountVectorizer(binary=True)
```

In [19]:

```
vocab=vect.vocabulary_
```

In [20]:

```
for key in sorted(vocab.keys()):
    print("{}:{}".format(key,vocab[key]))
```

```
character:0
engine:1
good:2
is:3
optical:4
recognition:5
significant:6
tessaract:7
```

In [21]:

```
print(vect.transform(["This is a good optical illusion"]).toarray())
```

```
[[0 0 1 1 1 0 0 0]]
```

In [22]:

```
print(vect.transform(corpus).toarray())
```

```
[[1 1 1 1 1 1 0 1]
 [1 0 0 1 1 1 1 0]]
```

In [23]:

```
from sklearn.metrics.pairwise import cosine_similarity
```

In [24]:

```
from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(vect.transform(["Google cloud is a good recognition engine"]
```

In [25]:

```
print(similarity)
```

```
[[0.67082039]]
```

In [ ]: