**filename: b9.py**

from math import sqrt

from math import log

from collections import Counter

from operator import itemgetter

def tf(kt,doc):

return (doc.count(kt))

def idf(kt,all\_docs):

num=0

for x in all\_docs:

if kt in x:

num=num+1

if num>0:

return round(float(log(float(len(all\_docs))/float(num))),3)

else:

return 0

def tfidf(kt,doc):

return (tf(kt,doc)\*idf(kt,all\_docs))

def cos\_sim(infile,docs,ktrms):

a=0

for x in ktrms:

a=a+tfidf(x,infile)\*tfidf(x,docs)

b=doclen(infile,ktrms)\*doclen(docs,ktrms)

if not b:

return 0

else:

return (round((a/b),3))

def doclen(doc,ktrms):

val=0

for x in ktrms:

val=val+pow(tfidf(x,doc),2)

return sqrt(val)

files=[]

all\_docs=[]

key\_terms=[]

documents=['doc1.txt','doc2.txt','doc3.txt','doc4.txt','doc5.txt','doc6.txt']

result=[['doc1.txt','animals'],['doc2.txt','animals'],['doc3.txt','animals'],['doc4.txt','sports'],['doc5.txt','sports'],['doc6.txt','sports']]

for x in documents:

files.append(open(x,'r').read())

for x in files:

all\_docs.append(x.lower().rstrip('\n'))

for x in all\_docs:

key\_terms=key\_terms+x.split()

key\_terms=set(key\_terms)

key\_terms=list(key\_terms)

filename=raw\_input("Enter test file: ")

inputfile=open(filename,'r').readline().lower()

cnt=0

for x in all\_docs:

result[cnt]=result[cnt]+[cos\_sim(inputfile,x,key\_terms)]

cnt=cnt+1

print result

k=3

sortedresult=sorted(result,key=itemgetter(2),reverse=True)

top\_k=sortedresult[:k]

top\_k[:]=(x for x in top\_k if x[2]!=0)

if len(top\_k)==0:

print "Does not match"

else:

class\_count=Counter(category for (document,category,value) in top\_k)

print class\_count

classification=max(class\_count,key=lambda cls:class\_count[cls])

print "Class of test file: ",classification

' ' '

TF: Term Frequency, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (aka. the total number of terms in the document) as a way of normalization:

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance. Thus we need to weigh down the frequent terms while scale up the rare ones, by computing the following:

IDF(t) = log\_e(Total number of documents / Number of documents with term t in it).

' ' '

**filename: doc1.txt**

Animals live on land and water.Land animals include cats, cows.Water animals include all types of fishes.

**filename: doc2.txt**

Animals can be classified as herbivorous which is plant eating, carnivorous which is flesh eating and omnivorous which is eating both.

**filename: doc3.txt**

All land animals have two eyes and ears.They have one tail.

**filename: doc4.txt**

Sports are all forms of usually competitive activity which,through casual or organised participation, aim to use, maintain or improve physical ability and skills.

**filename: doc5.txt**

Different sport have different rules.

**filename: doc6.txt**

Depending on the sport, every sport has different number of players in each team.

**filename: t.txt**

Dog is an omnivorous creature that lives on land and has one tail.

**filename: t2.txt**

Football is a competitive sport with eleven players.

**OUTPUT**

Amols-Air:b9 Darwin$ python b9.py

Enter test file: t.txt

[['doc1.txt', 'animals', 0.249], ['doc2.txt', 'animals', 0.162], ['doc3.txt', 'animals', 0.388], ['doc4.txt', 'sports', 0.067], ['doc5.txt', 'sports', 0.011], ['doc6.txt', 'sports', 0.164]]

Counter({'animals': 2, 'sports': 1})

Class of test file: animals