1 ASSIGNMENT NO: B9

Author: Ameeth Kanawaday
Roll No: 4430

2 Problem Definition

Using any similarity based techniques develop an application to classify text data. Perform pre-processing tasks as per requirement.

3 Learning Objectives:

1. To understand the concept of classification.
2. To understand the concept of data cleaning.

4 Theory

Classification:
Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify loan applicants as low, medium, or high credit risks.

Data Preprocessing:
Data pre-processing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values (e.g., Income: -100), impossible data combinations (e.g., Sex: Male, Pregnant: Yes), missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis.

If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The product of data pre-processing is the final training set. Kotsiantis et al. (2006) present a well-known algorithm for each step of data pre-processing.

Data Preprocessing Methods:

- Data Cleaning
- Data Integration
- Data Transformation

- Data Reduction

5 Related Mathematics

Let S be the solution perspective of the given problem.
The set S is defined as:
$S = \{ s, e, X, Y, F, DD, NDD | \varnothing_s \}$
Where,
s= Start point
s= Text data files unclassified.
where,
e= End point
e= Text data files classified based on similarity.
F= Set of main functions

F = $\{f_{prep}, f_{cos}, f_{class}\}$

f_{prep} :function to extract data in processable format.

f_{cos} :function to find cosine similarity of the text data file.

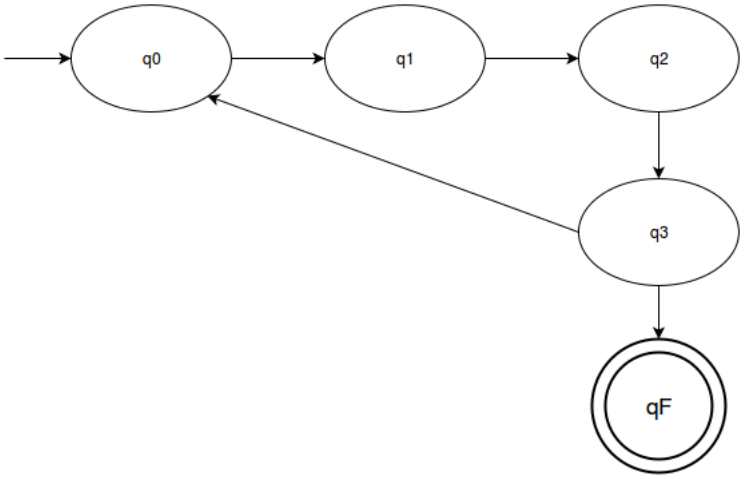
f_{class} : function to classify the files based on the similarity obtained.

X= Input Set.
$X = \{ f_1, ..., f_n \}$
where,
f_i= text files

Y = $\{C_1, C_2, ..., C_n\}$
where,
C_i : ith class of text relevance.
DD= set of deterministic data
NDD= set of non deterministic data

DD = $\{f_1, f_2, ..., f_n\}$
NDD = $\{C_1, C_2, ..., C_n\}$

6 State Diagram



q0 = read input text file
q1 = compute cosine similarity of the file
q2 = place file in the NN cluster
q3 = update overall similarity measure of the cluster.
qF = final state.

7 Program

PROGRAM

```
//filename: d1.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


//doc1.txt

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

//doc2.txt

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

//doc3.txt

Football is a competitive sport with eleven players.

//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.

//doc5.txt

Different sport have different rules.

//doc6.txt

Football is a competitive sport with eleven players.

//t.txt

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

//t1.txt

Football is a competitive sport with eleven players.


OUTPUT:
ameeth@ubuntu-16.0.4:~$ python d1.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', 0.012], ['doc6.txt', 'sports', 0.169]]
Counter({'animals': 2, 'sports': 1})
Class of test file:  animals
ameeth@ubuntu-16.0.4:~$ python d1.py
Enter test file: t2.txt
[['doc1.txt', 'animals', 0.051], ['doc2.txt', 'animals', 0.076],
 ['doc3.txt', 'animals', 0.034], ['doc4.txt', 'sports', 0.255],
 ['doc5.txt', 'sports', 0.072], ['doc6.txt', 'sports', 0.274]]
Counter({'sports': 2, 'animals': 1})
Class of test file:  sports
ameeth@ubuntu-16.0.4:~$
```

8 Conclusion

Hence, we studied various similarity based techniques and developed an application for text classification using data pre processing. We used cosine similarity to classify the data.