2017

Assignment 9

CS 432

MIRANDA SMITH

Question 1

Choose a blog or a newsfeed (or something similar with an Atom or RSS feed). Every student should do a unique feed, so please "claim" the feed on the class email list (first come, first served). It should be on a topic or topics of which you are qualified to provide classification training data. Find something with at least 100 entries (or items if RSS). Create between four and eight different categories for the entries in the feed:

Download and process the pages of the feed as per the week 12 class slides. Be sure to upload the raw data (Atom or RSS) to your github account. Create a table with 100 rows, like:

title	*classification*
Ric Ocasek - "Something To Grab For" (forgotten song)	*80s*
Weezer - "Pinkerton" (LP Review)	*alternative*
Schon & Hammer - "No More Lies" (forgotten song)	*80s*

etc. This is your "ground truth" (or "gold standard") data.

I chose the feed from http://sharkdivers.blogspot.com/

I got 4 rss feeds from the 'get_rss_feed.py' program. It went through each rss feed counting the number of entries and going to the next RSS feed until there were 100 entries. It just printed out the links and I downloaded them to rss_1.txt, rss_2.txt,

```
import feedparser


diving_rss="https://sharkdivers.blogspot.com/feeds/posts/default"


count = 0;
while count < 100:
    feed = feedparser.parse(diving_rss)
    print(diving_rss)

    for item in feed["items"]:
        count = count + 1


    fed = feed["feed"]
    for link in fed["links"]:
        if link["rel"] == "next":
            diving_rss = link["href"]
```

rss_3.txt and rss_4.txt.

I went through each RSS file with 'get_entry_links.py' and printed them out to a file.

```
import feedparser

rss_files = ["rss_1.txt" ,"rss_2.txt" ,"rss_3.txt" ,"rss_4.txt"]

file = open("Blog_Links.txt", 'w')

entries= {}
count = 0;
for rss in rss_files:
    feed = feedparser.parse(rss)

    for item in feed["items"]:
        file.write(item["link"])
        file.write("\n")
        count = count + 1

file.close()
```

I then manually went to each link and categorized the content. The categories went in the file ‘Blog_links_Categories.txt’

The categorizes I chose were

Shark Details

Trip Details

News

Conservation

I then created a program ‘getGroundTruthTable.py’ to create the required table. It just linked each category to the correct feed title and formatted the output.

ARTICLE	CLASSIFICATION
Get to know the Great White Sharks of Guadalupe	shark details
Do we need a shark cull at Reunion Island?	news
How many Great White Sharks are at Guadalupe Island?	shark details
Have a Sharky New Year!	news
Do shark repelling devices work?	news
Is shark diving beneficial for the sharks?	conservation
Is cage diving safe?	news
I was surrounded by bull sharks!	trip details
Update from Guadalupe Island	trip details
Shark feeding to be banned in US waters?	news
Another shark attack in Australia?	news
I was surrounded by bull sharks!	trip details
Shark "Expert" teaches how to survive a shark attack?	news
How do you choose the right shark diving operation?	news
What's it like to come face to face with a Great White Shark?	shark details
Update from our friends in Fiji	news
Who is more aggressive, a Bull Shark or a Great White?	shark details
Our friends in Fiji made it through the cyclone!	news
Is "BAD" really GOOD?	news
What is sustainable shark diving?	news
What kills fewer people than sharks each year?	news
Wanna dive with Bull Sharks?	trip details
Can our loving sharks be bad for them?	conservation
Breaching great white sharks at Guadalupe Island	news
How not to dive with sharks!	news

The entirety of the output in in ‘GroundTruthTable.txt’.

Question 2 & 3

Train the Fisher classifier on the first 50 entries (the "training set"), then use the classifier to guess the classification of the next 50 entries (the "test set"). Create a table with 50 rows. Assess the performance of your classifier in each of your categories by computing precision, recall, and F-measure. Use the "macro-averaged" label based method. For example, if you have 5 categories (e.g., 80s, metal, alternative, electronic, cover), you will compute precision, recall, and F-measure for each category, and then compute the average across the 5 categories. Repeat, but use the first 90 entries to train your classifier and the last 10 entries for testing.

I created the program 'getPredictedTable.py' to use the fisher method and create the required table.

I link the category with the entry of the RSS feed. Then train the fisher method with either the first 50 or 90 entries, depending on the run. I only use the title of the entry as the data, because that is how assignment 8 was done.

```
import textwrap
import feedparser
import docclass
import re
import math

rss_files = ["rss_1.txt" ,"rss_2.txt" ,"rss_3.txt" ,"rss_4.txt"]

file = open("Blog_Links_Categories.txt", 'r')
categories = file.readlines()
file.close()

actualData = {}
count = 0
for rss in rss_files:
        feed = feedparser.parse(rss)
        for item in feed["items"]:

            line = item["title"].strip()
            actualData[line] = categories[count]
            count = count +1


predictedData= {}

cl=docclass.fisherclassifier(docclass.getwords)
cl.setdb('mln.db')

#TRAINING
print("TRAINING")
count = 0
for rss in rss_files:
        feed = feedparser.parse(rss)
        for item in feed["items"]:
            #train with the first 50 or 90
            line = item["title"].strip()
            cl.train(line, actualData[line])
            if count == 89:
                break
            count = count +1
```

The next part of the program predicts the remaining entries and creates a table. It outputs a file called PredictedvsTruthTable50_50 or 90_10

```
#TESTING on the next 50 or 10

print("TESTING")
count = 0
for rss in rss_files:
    feed = feedparser.parse(rss)
    for item in feed["items"]:
        if count > 89:
            #predict the next 50 or 10
            line = item["title"].strip()
            predictedData[line] = cl.classify(line)
        count = count +1


with open("PredictedvsTruthTable90_10.txt" , 'w') as out:
    out.write ("{:<100} {:<20} {:<20}\n".format('ARTICLE','CLASSIFICATION','PRED CLASSIFICATION'))

    for a ,c  in predictedData.items():
        out.write ("{:<100} {:<20} {:<20}\n".format(a,actualData[a].strip(),c.strip()))
```

I used an excel work book 'PredictedvsTruthTable' to calculate the precision, recall and F-measure for both files.

For the 50-50 split:

category	false positive	false negative	True Positive	Precision	Recall	F-Measure
shark details	14	6	5	0.263157895	0.454545455	0.333333333
trip details	4	8	1	0.2	0.111111111	0.142857143
news	12	20	2	0.142857143	0.090909091	0.111111111
conservation	10	6	2	0.166666667	0.25	0.2
avg				0.193170426	0.226641414	0.196825397

For the 90-10 split:

category	false positive	false negative	True Positive	Precision	Recall	F-Measure
shark details	1	0	1	0.5	1	0.666666667
trip details	0	1	3	1	0.75	0.857142857
news	2	0	3	0.6	1	0.75
conservation	0	2	0	0	0	0
avg				0.525	0.6875	0.568452381

The 90-10 split is 2.9 times better than the 50 -50 split for F-Measure.