INTRO. TO WEB SCIENCE: CS 532: A9

Due on Monday, May 1, 2017

Dr. Nelson

Udochukwu Nweke

Contents

Problem 1

Listing 1: Training and Testing Blog Entries Code

```
import os, sys
import docclass
import feedparser
import feedfilter
from subprocess import check_output

def errorMsg():
     exc_type, exc_obj, exc_tb = sys.exc_info()
     fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
     print(fname, exc_tb.tb_lineno, sys.exc_info())

def FisherModel(trainingInputFileName, entriesXMLFileName, dbFileName, mode, maxItems):
     #input: trainingInputFileName.txt, entriesXMLFileName.xml,
     #mode is 'test' or 'train', 'getWord'|'getEntry'

     cl=docclass.fisherclassifier(docclass.getwords)
     '''
     if( getWordGetEntryMethod == 'getWord' ):
          cl=docclass.fisherclassifier(docclass.getwords)
     else:
          cl=docclass.fisherclassifier(feedfilter.entryfeatures)
     '''


     cl.setdb(dbFileName)
     feedfilter.getClassData(entriesXMLFileName, cl, trainingInputFileName, mode,
    maxItems)

def downloadBlogXML(blogUrl, outputFilename, countToProcess):

     try:

          output = check_output(['curl', '-s', blogUrl +
          'feeds/posts/default?max-results=' + str(countToProcess)])
          output = output.decode('utf-8')

          outputFile = open(outputFilename, 'w')
          outputFile.write(output)
          outputFile.close()
     except:
          print('Error parsing feed %s' % blogUrl)
          errorMsg()

#problem 1:
blogName = 'icovetthee'
blogUrl = 'http://www.' + blogName + '.com/'
xmlOutputFilename = './' + blogName + '.xml'
#download 10 feeds from blogUrl and save into xml file called blog.xml
#downloadBlogXML(blogUrl, xmlOutputFilename, 120)

```

Problem 1 continued on next page...

```

#problem 2 (training):
trainingCount = 50
dboutputFileName = blogName +'.db'
trainingInputfilename = 'Training-50Entries.txt'
#FisherModel(trainingInputfilename, xmlOutputFilename,
#dboutputFileName, 'train', trainingCount)

#problem 2 (testing):
trainingInputfilename = 'Testing-50Entries.txt'
#FisherModel(trainingInputfilename, xmlOutputFilename,
# dboutputFileName, 'test', trainingCount)




#problem 3 (training):
dboutputFileName = blogName +'.90.db'
#FisherModel('Training-90Entries.txt', xmlOutputFilename,
#dboutputFileName, 'train', 90)

#problem 3 (testing):
#FisherModel('Testing-10Entries.txt', xmlOutputFilename,
#dboutputFileName, 'test', 10)
```

Listing 2: Pci Code

```
import feedparser
import re
import os, sys


def getClassData(feed, classifier, inputFilename, mode, maxItemsToTestOrTrain,
wordOrEntry='word'):

 if( len(feed) > 0 and len(inputFilename) > 0 and (mode == 'train' or mode == 'test')
 and maxItemsToTestOrTrain > 0 and (wordOrEntry == 'word' or wordOrEntry == 'entry')):

  #inputFilename: <title, titleText, classLabel>
  try:
   inputFile = open(inputFilename, 'r')

   if( mode == 'test' ):
    prefix = inputFilename.split('.')[0]
    outputFile = open(prefix+'Predictions.txt', 'w')

   lines = inputFile.readlines()
   inputFile.close()
   #first line is schema
   del lines[0]
   print( len(lines), 'lines read from ' + inputFilename)
  except:
   exc_type, exc_obj, exc_tb = sys.exc_info()
   fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
```

```
      print((fname, exc_tb.tb_lineno, sys.exc_info() ))
      return

    if( mode == 'test' ):
      outputFile.write('TITLE <> PROB <> PREDICTED-LABEL <> ACTUAL-LABEL\n')

    # Get feed entries and loop over them
    f=feedparser.parse(feed)
    count = 1
    for entry in f['entries']:

      for l in lines:

        titleContentLabel = l.split('<>')

        title = titleContentLabel[0].strip()
        #print('\ttitle:', title)
        summary = titleContentLabel[1].strip()
        actualClassLabel = titleContentLabel[2].strip()

        if( title.lower() == entry['title'].strip().lower() ):
          fulltext='%s\n%s' % (entry['title'],entry['summary'])

          if( mode == 'train' ):
            #training get the correct category and train on that

            if( wordOrEntry == 'word' ):
              classifier.train(fulltext, actualClassLabel)
            else:
              classifier.train(entry, actualClassLabel)

            print( '...training count:', count)
          else:
            #testing: guess the best guess at the current category
            try:
              if( wordOrEntry == 'word' ):
                prediction = str(classifier.classify(fulltext))
              else:
                prediction = str(classifier.classify(entry))

              classPredictionProbability = classifier.getGlobalCProbValue()
              print( '...testing count', count)
              print( title + ' <> ' + str(classPredictionProbability) + ' <> '
              + prediction + ' <> ' + actualClassLabel )
              outputFile.write(title + ' <> ' + str(classPredictionProbability) + ' <> '
              + prediction + ' <> ' + actualClassLabel + '\n')
            except:
              exc_type, exc_obj, exc_tb = sys.exc_info()
              fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
              print((fname, exc_tb.tb_lineno, sys.exc_info() ))
              print( '...skipping', count)

```

```
     if(count == maxItemsToTestOrTrain):
      print( '...max items reached, closing')

      if( mode == 'test' ):
       outputFile.close()

      return

     count += 1

  if( mode == 'test' ):
   outputFile.close()
```

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

examples:

work, class, family, news, deals liberal, conservative, moderate, libertarian sports, local, financial, national, international, entertainment metal, electronic, ambient, folk, hip-hop, pop

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 -    80s
``Something To Grab
For''  (forgotten song)

Weezer - ``Pinkerton''  alternative
(LP Review)

Schon \& Hammer -  80s
``No More Lies''
(forgotten song)
```

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

Solution 1:

1. I considered a blog with at least 100 entries and content . With this as my target, I claimed `http://www.icovetthee.com/`.

2. After going through (`http://www.icovetthee.com/`.) I classified the entries into four categories and they are: beauty, lifestyle, style and miscellaneous (misc)

3. In order to extract the Atom feed of the blog, I used *downloadBlogXML()* in listing 1 and saved the xml file into *icovetthee.xml* . Table 1 shows the blog tiles and the blog classifications

Problem 2

Listing 3: Training and Testing Data

```python
import os, sys

from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score

def getPredictActualLabels(inputFileName):

    listOfPredictedLabels = []
    listOfActualLabels = []

    try:
        inputFile = open(inputFileName, 'r')
        lines = inputFile.readlines()

        del lines[0]
        print( len(lines), 'lines read from ' + inputFileName )
        inputFile.close()
    except:
        exc_type, exc_obj, exc_tb = sys.exc_info()
        fname = os.path.split(exc_tb.tb_frame.f_code.co_filename)[1]
        print(fname, exc_tb.tb_lineno, sys.exc_info() )

    for l in lines:

        predictedAndActualLabel = l.split(' <> ')
        if( len(predictedAndActualLabel) > 1 ):

            predictedAndActualLabel = predictedAndActualLabel[-2:]

            predictedLabel = predictedAndActualLabel[0].strip()
            actualLabel = predictedAndActualLabel[1].strip()

            listOfPredictedLabels.append(predictedLabel)
            listOfActualLabels.append(actualLabel)

    return listOfPredictedLabels, listOfActualLabels
```

Problem 2 continued on next page. . .

```
def main(predictionFilename):

    infile = open('./predictionLabels.txt', 'r')
    labels = infile.read()
    infile.close()

    labels = labels.split(', ')
    print('\tlabels:', labels)

    listOfPredictedLabels, listOfActualLabels = getPredictActualLabels
    (predictionFilename)

    confusionMatrix = confusion_matrix( listOfActualLabels, listOfPredictedLabels,
    labels=labels )
    precision = precision_score( listOfActualLabels, listOfPredictedLabels,
    labels=labels, average='macro' )
    recall = recall_score( listOfActualLabels, listOfPredictedLabels,
    labels=labels, average='macro' )
    f1 = f1_score( listOfActualLabels, listOfPredictedLabels, labels=labels,
   average='macro' )

    print('\nconfusion matrix:')
    print( confusionMatrix )

    print('\nprecision:')
    print( precision )

    print( '\nrecall:' )
    print( recall )

    print( '\nf1' )
    print( f1 )

if __name__ == "__main__":

    if( len(sys.argv) != 2 ):
        print('\tMissing prediction input filename')
        print('\tE.g python eval.py Testing-50Predictions.txt')
    else:
        main( sys.argv[1] )
```

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, like

```
title     actual    predicted
-----     ------    ---------
Donnie Iris -     80s    80s
``Ah! Leah!``
(Forgotten Song)

Black Sabbath -   metal   metal
``Vol. 4'' (LP Review)
```

```
Catherine Wheel -   alternative metal
``Ferment'' (LP Review)
```

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, as per:
`http://stats.stackexchange.com/questions/21551/how-to-compute-precision-recall-for-multicl`
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.

Solution 2:

1. In order to train the fish classifier on the first 50 entries and use the classifier to guess the next 50 entries, I created two files from *BlogMe.txt*. *Training-50Entries.txt* and *Testing-50Entries.txt*. *Training-50Entries.txt* is my training set and *Testing-50Entries.txt* is my test set.

2. I used *FisherModel(trainingInputfilename, xmlOutputFilename, dboutputFileName, ‘train’, trainingCount)* in listing 1 to train my first 50 entries (*Training-50Entries.txt*) and the result of the training is written in *icovetthee.db*

3. I used *FisherModel(trainingInputfilename, xmlOutputFilename, dboutputFileName, ‘test’, trainingCount)* in listing 1 to test (*Testing-50Entries.txt*) and the result of test is written in *Testing-50EntriesPredictions.txt*.

4. I used *getPredictActualLabels* in listing 3 to test *Testing-50Entries.txt* in order to predict the category of each blog tile. The result of the test is saved in *Testing-50EntriesPredictions.txt* and Table 2 shows the result of testing 50 entries with prediction.

5. In order to assess the performance of my classifier for each category, I computed precision, recall, and F-measure of each of the blog categories. This was achieved by using *main(predictionFilename)* in listing 3. Table 3 Shows the precision, recall, and F-measure for the blog categories.

Problem 3

Repeat question #2, but use the first 90 entries to train your classifier and the last 10 entries for testing.

Solution 3:

1. I split the 100 entries bog from *BlogMe.txt* into 10 and 90 entries. I used the first 90 entires as my training set and it is saved in *Training-90Entries.txt* and the remaing 10 entries, my test set and it is saved in *Testing-10Entries.txt*

2. I used *FisherModel(trainingInputfilename, xmlOutputFilename, dboutputFileName, ‘train’, trainingCount)* in listing 1 to train *Training-90Entries.txt* entries and the result of the training is written in *icovetthee.90.db*

3. I used *FisherModel(trainingInputfilename, xmlOutputFilename, dboutputFileName, ‘test’, trainingCount)* in listing 1 to test and the result of the is written in *Testing-10EntriesPredictions.txt*.

4. In order to assess the performance of my classifier for each category, I computed precision, recall, and F-measure of each of the blog categories. This was achieved by using *main(predictionFilename)* in listing 3.

Problem 3 continued on next page. . .

Table 4 Shows the precision, recall, and F-measure for the blog categories.

Observation: Training 90 entries and Testing 10 entries showed a higher precision presumably because we had more training data.

Table 1: 100 Blog Entries

Item	Title	Classification
1	How I Style My Hair: Easy Laid Back Waves	beauty
2	My Boxing Day Sales Picks	style
3	What I'll Be Drinking This Christmas Eve	misc
4	What's on My Christmas List This Year	style
5	My Five Favourite Products Of The Year	beauty
6	Silk For Your Skin	beauty
7	The Three Best Apps For Instagram	misc
8	Gucci Bamboo	beauty
9	Summer Denim	style
10	Starting Your Day With The Right Skincare	beauty
11	Wearing White In Winter	style
12	A Swoon Worthy Diptyque Candle	misc
13	Hello, It's I Covet Thee 4.0!	misc
14	Taking You Through My Skincare Routine	beauty
15	utumn Smokey Make Up Using the Lorac Pro Palette	beauty
16	This Week: Feeling Autumnal	lifestyle
17	What's In My Bag: Autumn Edition	lifestyle
18	Suede on Black	style
19	Autumn Style Picks	style
20	My Most Used Products in September	beauty
21	My Go-To Budget Friendly Make Up Look	misc
22	What I Picked Up From The US	beauty
23	A Lazy Girl's Guide to Tanning	beauty
24	August Favourites	beauty
25	Charlotte Tilbury Magic Foundation: An Exciting New Launch	beauty
26	A Simple Step That Gets Me Up In The Morning	misc
27	The Date Night: A Red Inspired Get Ready With Me	lifestyle
28	The Week: Catching Up	lifestyle
29	July Favourites	beauty
30	This Week: Seven Days in L.A.	misc
31	Mid-Year Beauty Round Up: Make Up & Skincare Favourites	beauty
32	New Skincare Additions	beauty
33	Packing For a Week in L.A.!	lifestyle
34	What's In My Bag: Whistle Fleet Tote	lifestyle
35	Make Up Haul & First Impressions	misc
36	Clinique Pop Lips	beauty
37	Two Kiehl's Products I'm Loving	beauty
38	June Favourites	beauty
39	This Week: The Wimbledon Dress	style
40	#ICovetJune Round Up: Part Four	lifestyle
41	Birthday Make Up!	beauty
42	This Week: Turning Twenty Three	lifestyle
43	#ICovetJune Round Up: Part Three	lifestyle
44	White Out	style

Item	Title	Classification
45	Get Un-Ready With Me: After Party Night Time Routine	beauty
46	#ICovetJune Round Up: Part Two	lifestyle
47	Summer Bronze Smokey Eye	misc
48	This Week: The Breakfast Club, Brighton	misc
49	#ICovetJune Round Up: Part One	lifestyle
50	May Favourites & Exciting Announcement!	misc
51	This Week: Sun, Sea & Tapas	style
52	Five Years of Topshop Beauty	beauty
53	Cherry Blossoms & Tulle	style
54	Current Skincare Favourites & Morning/Evening Routine	beauty
55	April Favourites	misc
56	This Week: A Magical Disney Weekend!	lifestyle
57	I Went to Sephora and Only Bought One Thing	misc
58	Boots Beaty Haul & First Impressions	misc
59	This Week: Bonjour Paris!	lifesyle
60	Budget Dupes & Drugstore Beauty Alternatives	misc
61	Nutella Stuffed Chocolate Chip Cookies	misc
62	I Covet Thee Turns Four!	lifestyle
63	March Favourites	misc
64	My Everyday Make Up Bag & Routine	misc
65	This Week: Charlotte Tilbury X Norman Parkinson	lifesyle
66	The Best Budget Micellar Water	beauty
67	This Week: The South's Best Mac & Cheese	lifesyle
68	This Week: The South's Best Mac & Cheese	misc
69	February Favourites	misc
70	Hair Chat: Favourite Products & Styling Routine	misc
71	Topshop, Asos & Missguided Haul	misc
72	A Little Trip to Paris	misc
73	January Favourites	misc
74	he Best Healthy Cookies	misc
75	Disappointing Products	misc
76	This Week: Lunch at Jamie's	misc
77	Drugstore Make Up Favourites	misc
78	New In: Three from the Drugstore	lifestyle
79	Minimal Make Up Routine	misc
80	Wear It, Beat It	style
81	This Week: Meet Up's & Halloumi Overload	misc
82	Three New Year's Lifestyle Resolutions	lifestyle
83	Best of Beauty in 2014	beauty
84	Summing Up Vlogmas	misc
85	The Post-NYE Saviour	misc
86	December Favourites	misc
87	The Best of I Covet Thee in 2014	style
88	Beauty Sales Picks	beauty
89	The Pre-Christmas Pamper #ICovetChristmas	misc

Item	Title	Classification
90	Christmas Q&A With Suzie! #ICovetChristmas	misc
91	The Christmas Jumper #ICovetChristmas	misc
92	Christmas Party Make Up #ICovetChristmas	misc
93	Serozinc Comes to the UK	misc
94	Three Winter Coats	misc
95	Winter Skincare Staples	misc
96	November Favourites	misc
97	What's On My Christmas List	style
98	Weekend Vlog: Christmas Markets & Crepes	misc
99	Jennifer Lawrence Inspired Drugstore Make Up Look	beauty
100	Autumn Beauty Haul	beauty

Table 2: Test Set

Item	Title	Actual	Predicted
1	This Week: Sun, Sea & Tapas	style	style
2	Five Years of Topshop Beauty	beauty	beauty
3	Cherry Blossoms & Tulle	style	style
4	Current Skincare Favourites & Morning/Evening Routine	beauty	beauty
5	April Favourites	misc	beauty
6	This Week: A Magical Disney Weekend!	lifestyle	misc
7	I Went to Sephora and Only Bought One Thing	misc	beauty
8	Boots Beaty Haul & First Impressions	misc	beauty
9	This Week: Bonjour Paris!	lifesyle	beauty
10	Budget Dupes & Drugstore Beauty Alternatives	misc	beauty
11	Nutella Stuffed Chocolate Chip Cookies	misc	misc
12	I Covet Thee Turns Four!	lifestyle	misc
13	March Favourites	misc	beauty
14	My Everyday Make Up Bag & Routine	misc	beauty
15	This Week: Charlotte Tilbury X Norman Parkinson	lifesyle	beauty
16	The Best Budget Micellar Water	beauty	beauty
17	This Week: The South's Best Mac & Cheese	lifesyle	misc
18	This Week: The South's Best Mac & Cheese	misc	misc
19	February Favourites	misc	beauty
20	Hair Chat: Favourite Products & Styling Routine	misc	beauty
21	Topshop, Asos & Missguided Haul	misc	style
22	A Little Trip to Paris	misc	beauty
23	January Favourites	misc	beauty
24	he Best Healthy Cookies	misc	beauty
25	Disappointing Products	misc	beauty
26	This Week: Lunch at Jamie's	misc	misc
27	Drugstore Make Up Favourites	misc	misc
28	New In: Three from the Drugstore	lifestyle	beauty
29	Minimal Make Up Routine	misc	beauty
30	Wear It, Beat It	style	beauty
31	This Week: Meet Up's & Halloumi Overload	misc	beauty
32	Three New Year's Lifestyle Resolutions	lifestyle	beauty
33	Best of Beauty in 2014	beauty	beauty
34	Summing Up Vlogmas	misc	misc
35	The Post-NYE Saviour	misc	beauty
36	December Favourites	misc	beauty
37	The Best of I Covet Thee in 2014	style	beauty
38	Beauty Sales Picks	beauty	beauty
39	The Pre-Christmas Pamper #ICovetChristmas	misc	beauty

Item	Title	Actual	Predicted
40	Christmas Q&A With Suzie! #ICovetChristmas	misc	beauty
41	The Christmas Jumper #ICovetChristmas	misc	beauty
42	Christmas Party Make Up #ICovetChristmas	misc	misc
43	Serozinc Comes to the UK	misc	misc
44	Three Winter Coats	misc	beauty
45	Winter Skincare Staples	misc	beauty
46	November Favourites	misc	misc
47	What's On My Christmas List	style	beauty
48	Weekend Vlog: Christmas Markets & Crepes	misc	misc
49	Jennifer Lawrence Inspired Drugstore Make Up Look	beauty	misc
50	Autumn Beauty Haul	beauty	misc

Table 3: Precision, Recall and F1 for 50 Blog Entries Categories

Item	Class	Precision	Recall	F1
1	Beauty	0.161290322581	0.714285714286	0.263157894737
2	style	0.666666666667	0.400000000000	0.500000000000
3	lifestyle	0.000000000000	0.000000000000	0.000000000000
4	misc	0.666666666667	0.322580645161	0.434782608696

Table 4: Precision, Recall and F1 for 10 Blog Entries Categories

Item	Class	Precision	Recall	F1
1	Beauty	1.000000000000	0.333333333333	0.500000000000
2	style	0.400000000000	1.000000000000	0.571428571429
3	misc	0.000000000000	0.000000000000	0.000000000000

References

[1] Blog Time Now. http://blogtimenow.com/blogging/find-blogger-blog-id-post-id-unique-id-number/. Accessed: 2017-10-04.

[2] Toby Segaran. Programming Collective Intelligence, 2007.