CS 595: Assignment #10

Due on Thursday, December 11, 2014

Dr. Nelson 4:20pm

Holly Harkins

Contents

Problem 1

```
Choose a blog or a news feed (or something similar as long as it has
an Atom or RSS feed).  It should be on a topic or topics of which you
are qualified to provide classification training data.  In other words,
choose something that you enjoy and are knowledgeable of.  Find a feed
with at least 100 entries.

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.



Answer:
Python on my laptop was not working so I had to complete this assignment
without my results.

I chose blog http://taylor-swift-love.blogspot.com/.  Using a curl
statement I would grab the first 100 entries.  I used the categories
below for the first 100 entries in the feed and classified them
manually.

international
fashion
awards
shows

http://www.taylor-swift-love.blogspot.com/feeds/posts/default?max-results=100

In Assignment10.py, the entries are looped through which generates the word counts for
each entry. Train the classifier with my chosen categories. Identifies a category for
each of them and outputs results.
```

Listing 1: Assigment 10 Python

```python
# -*- coding: utf-8 -*-
import feedparser
import re
import sys
import math
from operator import itemgetter

def getwords(doc):
  splitter=re.compile('\\W*')
  doc=re.compile(r'<[^>]+>').sub('',doc)
  words=[s.lower() for s in splitter.split(doc)
          if len(s)>2 and len(s)<20]
  word=[]
  for W in dict([(w,1) for w in words]):
     word.append(W)
  return word

class classifier:
  def __init__(self,getfeatures,filename=None):
    self.fc={}
    self.cc={}
    self.getfeatures=getfeatures

  def incf(self,f,cat):
    self.fc.setdefault(f,{})
    self.fc[f].setdefault(cat,0)
    self.fc[f][cat]+=1

  def incc(self,cat):
    self.cc.setdefault(cat,0)
    self.cc[cat]+=1

  def fcount(self,f,cat):
    if f in self.fc and cat in self.fc[f]:
     return float(self.fc[f][cat])
    return 0.0

  def catcount(self,cat):
    if cat in self.cc:
     return float(self.cc[cat])
    return 0

  def categories(self):
    return self.cc.keys( )

  def train(self,item,cat):
    features=self.getfeatures(item)
    for f in features:
      self.incf(f,cat)
    self.incc(cat)

  def fprob(self,f,cat):
    if self.catcount(cat)==0: return 0
    return self.fcount(f,cat)/self.catcount(cat)

  def weightedprob(self,f,cat,prf,weight=1.0,ap=0.5):
    basicprob=prf(f,cat)
    totals=sum([self.fcount(f,c) for c in self.categories()])
    bp=((weight*ap)+(totals*basicprob))/(weight+totals)
    return bp

class fisherclassifier(classifier):
  def cprob(self,f,cat):
    clf=self.fprob(f,cat)
    if clf==0: return 0
    freqsum=sum([self.fprob(f,c) for c in self.categories()])
    p=clf/(freqsum)
    return p

  def fisherprob(self,item,cat):
    p=1
    features=self.getfeatures(item)
    for f in features:
      p*=(self.weightedprob(f,cat,self.cprob))
    fscore=-2*math.log(p)
    return self.invchi2(fscore,len(features)*2)

def invchi2(self,chi, df):
    m = chi / 2.0
    sum = term = math.exp(-m)
    for i in range(1, df//2):
        term *= m / i
        sum += term
    return min(sum, 1.0)

  def __init__(self,getfeatures):
    classifier.__init__(self,getfeatures)
    self.minimums={}

  def setminimum(self,cat,min):
    self.minimums[cat]=min

  def getminimum(self,cat):
    if cat not in self.minimums: return 0
    return self.minimums[cat]

  def classify(self,item,default=None):
    best=default
    max=0.0
    for c in self.categories():
      p=self.fisherprob(item,c)
      if p>self.getminimum(c) and p>max:
        best=c
        max=p
     print str(round(p,4))+"&"
     return best

def entryfeatures(entry):
  splitter=re.compile('\\W*')
  f={}
```

```
  titlewords=[s.lower() for s in splitter.split(entry[’title’])
          if len(s)>2 and len(s)<20]
  for w in titlewords: f[’Title:’+w]=1

  summarywords=[s.lower() for s in splitter.split(entry[’summary’])
          if len(s)>2 and len(s)<20]

  uc=0
  for i in range(len(summarywords)):
    w=summarywords[i]
    f[w]=1
    if w.isupper(): uc+=1

    if i<len(summarywords)-1:
      twowords=’ ’.join(summarywords[i:i+1])
      f[twowords]=1

  return f

def main():
  cl=classifier(getwords)
  cl.train(’foreign, world, japan’,’international’)
  cl.train(’model, perfume, clothes’,’fashion’)
  cl.train(’award, nominated, nominee’,’awards’)
  cl.train(’tour, schedule, concert’,’shows’)
  print cl.categories()
  f=feedparser.parse(’feedlist.xml’)
  i=0
  manul_entry={}
  manul_sumry={}
  second_fifty_entry={}
  for entry in f[’entries’][0:100]:

   title=entry[’title’].encode(’utf-8’)
   Sumry=’%s\n%s’ % (entry[’title’],entry[’summary’])

   i+=1
   Dic=getwords(Sumry)
   categ=’international’
   I_total=0.0
   A_total=0.0
   F_total=0.0
   S_total=0.0
   for w in Dic:
    I_total+=cl.fcount(w,’international’)
    A_total+=cl.fcount(w,’awards’)
    F_total+=cl.fcount(w,’fashion’)
    S_total+=cl.fcount(w,’shows’)
   value = max(I_total,A_total,F_total,S_total)
   if value==F_total:
    categ=’fashion’
   if value==S_total:
    categ='shows'
   if value==A_total:
    categ='awards'
   if value==I_total:
    categ='international'
   manul_entry[title]=categ
   manul_sumry[title]=Sumry
   print str(i)+': '+title+"\t\t"+categ

  cl=fisherclassifier(getwords)
  for key,value in manul_sumry.iteritems():
    cl.train(key,manul_entry[key])

  for entry in f['entries'][50:100]:
   title=entry['title'].encode('utf-8')
   T_Sumry='%s\n%s' % (entry['title'],entry['summary'])
   i+=1
   print title+"&"
   predicat=str(cl.classify(T_Sumry))
   print predicat+"&"+manul_entry[title]+"\\\\"
   cl.train(T_Sumry,predicat)
   actual=manul_entry[title]

main();
```

Problem 2

Manually classify the first 50 entries, and then classify (using the fisher classifier) the remaining 50 entries. Report the cprob() values for the 50 titles as well. From the title or entry itself, specify the 1-, 2-, or 3-gram that you used for the string to classify. Do not repeat strings; you will have 50 unique strings. For example, in these titles the string used is marked with *s:

Rachel Goswell - "Waves Are Universal" (LP Review)
The *Naked and Famous* - "Passive Me, Aggressive You" (LP Review)
Negativland - "Live at Lewis's, Norfolk VA, November 21, 1992" (concert)
Negativland - "*U2*" (LP Review)

Note how "Negativland" is not repeated as a classification string.

Create a table with the title, the string used for classification, cprob(), predicted category, and actual category.

Answer:
From Assignment10.py, the function fisherclassifier classify the remaining 50 entries. It will get the predicted category for each entry, from fisher classifier, and train the classifier with each category predication.

fisherclassifier function- The frequency of this feature in all the categories. The probability is the frequency in this category divided by the overall frequency. Loop through looking for the best result and make sure it exceeds its minimum.

fisherprob funtion-Multiply all the probabilities together. Take the natural log and multiply by -2. Use the Inverse chi-squared function to get a probability.

Table would be displayed here showing Classifier Data: title, classifier, predicted, actual, and the cprob() .

Problem 3

Assess the performance of your classifier in each of your categories by computing precision, recall, and F1. Note that the definitions of precisions and recall are slightly different in the context of classification; see:

http://en.wikipedia.org/wiki/Precision_and_recall#Definition_.28classification_context.29

and

http://en.wikipedia.org/wiki/F1_score

Answer: The results of precision and recall would be displayed here. Depending on if the entries were categorized correctly would determine if the prediction accuracy is the accuracy based off its prediction, or that category versus the actual. False positives could be determined if incorrect classifiers were used.

	precision	recall
international	0	0
fashion	0	0
awards	0	0
shows	0	0

Table 1: Preformance Measures

Problem 4

Redo questions 2 & 3, but with manually train 90 entries and then classify the remaining 10.

Then redo questions 2 & 3, but with the extensions on slide 26 and pp. 136--138. Fully discuss the changes you’ve made.

Which method (more training vs. better features) gave better improvement over your baseline? Why do you think that is?

References

[1] Precision and recall. http://en.wikipedia.org/wiki/Precisionandrecall

[2] Segaran, T. Programming Collective Intelligence: Building Smart Web 2.0 Applications. O'Reilly, 2007.