Amara Naas (anaas)
CS595 — Fall 2013 **Assignment 10** 12/13/2013

## Problem 1

In my approach to work this assignment I used some functions which available at chapter six of our text book[1] and used the `http://www.salary.com/personal/feeds/slideshow_feed.xml` from [2]. I save the xml page in a file called *feedlist.xml*[1]. I Created four different categories for the first 100 entries in the feed and classified them manually base on the following classification:

- salary: pay, paid, cash.
- job: employee, employer, work, worker, coworker, boss.
- career: idea, gift, career, goal, dream, suggestion.
- resume: train, profession, instruct, teach, application, expert.

In first program *Q1.py* I used *getwords* function, the class *class classifier* from [1] and the *feedlist.xml*[1] file. the program dose the following:

- Looping over the feed to read the entries from feedlist.xml and generates the word counts for each entries.

```
def getwords(doc):
  splitter=re.compile('\\W*')
  #-----------1 from assignment 9 ----------
  ## Remove all the HTML tags
  doc=re.compile(r'<[^>]+>').sub('',doc)
  #-----------------2-----------------
  # Split the words by non-alpha characters
  words=[s.lower() for s in splitter.split(doc)
         if len(s)>2 and len(s)<20]
  word=[]
  # Return the unique set of words only
  for W in dict([(w,1) for w in words]):
       word.append(W)
  return word
```

- Train the classifier with my chosen categories.

```
def main():
  cl=classifier(getwords)
  cl.train('salary pay paid cash','salary')
  cl.train('job employee employer work worker coworker boss','job')
  cl.train('idea gift career goal dream suggestion','career')
  cl.train('resume train profession instruct teach application
     expert','resume')
```

[1]File uploaded to github

- Loop over the entries and manually identifies a category for each of them and print the result which saved in a file called *ActualCateg.txt*.

```
f=feedparser.parse('feedlist.xml')
i=1
for entry in f['entries'][0:100]:
 title=entry['title'].encode('utf-8')#.replace("'","")
 Sumry='%s\n%s' % (entry['title'],entry['summary'])
 i+=1
 Dic=getwords(Sumry)
 categ='salary'
 S_total=0.0
 J_total=0.0
 C_total=0.0
 R_total=0.0
 for w in Dic:
  S_total+=cl.fcount(w,'salary')
  J_total+=cl.fcount(w,'job')
  C_total+=cl.fcount(w,'career')
  R_total+=cl.fcount(w,'resume')
 value = max(S_total,J_total,C_total,R_total)
 if value==C_total:
  categ='career'
 if value==R_total:
  categ='resume'
 if value==J_total:
  categ='job'
 if value==S_total:
  categ='salary'
 first_fifty_entry[title]=categ
 print title+"\t\t"+categ
print first_fifty_entry
```

## Problem 2

I add class *fisherclassifier* from [1] to the program in question 1 and run the program from file call *Q2.py*[1]. This program will do the same as in question 1 and will do the following additional tasks:

1. The first fifty entries with its categories used to train the classifier.

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

2. Looping over the second fifty entries and get the predicted category for each entry from fisher classifier and train the classifier with each category prediction. Print the result of both, the actual and the predicted category, and save them in file called *PredictActualCatprob.txt*.

```
for entry in f['entries'][50:100]:
# Print table of entries' title
 title=entry['title'].encode('utf-8')#.replace("'","")
 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)
```

Table 1 shows the results of title, the cprob(), predicted category, and actual category.

| title | predicted category | actual category | cprob() |
|---|---|---|---|
| The Job Hunter's Guide to Social Media | job | job | 0.987 |
| 7 Salary Negotiation Tips for Women - How to Get Ahead Without Negative Feedback | salary | salary | 0.995 |
| Gaming Your Workday - How Video Games Can Help Your Career | job | job | 0.937 |
| Horrible Bosses - Real-Life Tales of Workplace Terror | job | job | 0.986 |
| Going Paperless: Is it time to buy a tablet? | salary | job | 0.313 |
| 15 More Must-Read Business Books That Could Change Your Life | salary | salary | 0.987 |
| Job Interview For Dads | job | job | 0.836 |
| Salary.com's 2011 Dad Salary Survey - How Much Is Your Dad Worth? | job | job | 0.992 |
| 2011 Best Places to Work for Recent Grads | job | job | 0.947 |
| Spring Clean Your Way to Career Success: 12 Orderly Office Tips | job | job | 0.993 |
| Going Green: 12 Awesome Earth-Friendly Jobs | job | career | 0.736 |
| 10 Ways to Use Twitter to Get Recruiters' Attention | job | job | 0.913 |
| What's Mom Worth? It's More Than You Think! | salary | salary | 0.996 |
| 13 Must-Read Business Books That Could Change Your Life | salary | job | 0.141 |
| Make Unemployment Work for You: 14 Dos and Don'ts | job | job | 0.942 |
| Dos and Don'ts for Creating Your Online Presence | job | job | 0.929 |
| 14 Common Job Hunting Blunders | job | job | 0.946 |
| Right and Wrong Answers to 8 Classic Interview Questions | job | job | 0.957 |
| From Forbes: Big Achievers Share The Greatest Risks They Ever Took | salary | salary | 0.998 |
| It's a Dog's Life: Six Animal Careers We'd Love to Have | job | job | 0.973 |
| First Days on the Job: 15 Ways to Make a Great Impression | job | job | 0.939 |
| Everything I Know about Business I Learned from My Pet | salary | salary | 0.998 |
| Avoid Foreign Faux Pas: 14 Office Customs around the World | salary | salary | 0.989 |
| It's Inappropriate, but Is It Illegal? 7 Iffy Office Scenarios | salary | salary | 0.990 |

## Problem 3

I used kcluster function from [1] to solve problem 3 which operates based on an algorithm called K-Means Clustering. This algorithm breaks the data into distinct groups because it is told in advance how many distinct clusters to generate. this function dose the following:

Table 1 shows the results when k is 5, 10, and 20.

| | precision | recall |
|---|---|---|
| salary | 10/10 | 0 |
| job | 28/33 | 4/28 |
| career | 0 | 1/4 |
| resume | 0 | 1/6 |

Table 2: Preformance Measures

## Problem 4

For Q4 I used the function *entryfeatures* obtained from [1]. It works on types more complicated than just strings where it takes the whole entry. In this function pairs of consecutive words are added as features. This function can be run from file called *Q4.py*[2].

[2]File uploaded to github

## References

[1] T. Segaran, CHAPTER 6 Document Filtering, pp. 29–53 in: “Programming Collective Intelligence”, O’REILLY, Aug. 2007.

[2] `http://www.salary.com/`.