Web Science cs532-s16

Assignment 10 Report

Dr. Michael L. Nelson

By: Huan Huang
04/30/2016

Problem 1

```
Using the data from A8:

- Consider each row in the blog-term matrix as a 500 dimension vector,
corresponding to a blog.

- From chapter 8, replace numpredict.euclidean() with cosine as the
distance metric.  In other words, you'll be computing the cosine between
vectors of 500 dimensions.

- Use knnestimate() to compute the nearest neighbors for both:

http://f-measure.blogspot.com/
http://ws-dl.blogspot.com/

for k={1,2,5,10,20}.
```

Answer

For this problem, I had some help from fellow classmate Ryan. Using data from assignment 8, I use the blog name "F-Measure" and "Web Science and Digital Libraries Research Group" to calculate the distance between these two blogs and every other blogs in the data file. It is done by calling function knnestimate(), which calls for getdistance() function. Getdistance() function will call for cosine() function to calculate distance between 2 vectors and return the result to knnestimate(). Knnestimate() then return the closest vectors based on the k values.

```
K Nearest Neighbors for F-Measure:
When k = 1
Pithy Title Here
When k = 2
Pithy Title Here
The Jeopardy of Contentment
When k = 5
Pithy Title Here
The Jeopardy of Contentment
theindiefriend
DaveCromwell Writes
The Listening Ear
When k = 10
Pithy Title Here
The Jeopardy of Contentment
theindiefriend
DaveCromwell Writes
The Listening Ear
The Girl at the Rock Show
Kid F
The Power of Independent Trucking
My Name Is Blue Canary
Steel City Rust
When k = 20
Pithy Title Here
The Jeopardy of Contentment
theindiefriend
DaveCromwell Writes
The Listening Ear
The Girl at the Rock Show
Kid F
The Power of Independent Trucking
My Name Is Blue Canary
Steel City Rust
Blog Name Pending
A Wife's Tale
The Ideal Copy
www.doginasweater.com Live Show Review Archive
The World's First Internet Baby
Rants from the Pants
I/LOVE/TOTAL/DESTRUCTION
Tremagazine
isyeli's
.

```

```python
#! /usr/bin/python

import math

def cosine(v1, v2):
  sumxx, sumyy, sumxy = 0 ,0 ,0
  for i in range(len(v1)):
    x = v1[i]; y = v2[i]
    sumxx += float(x)*float(x)
    sumyy += float(y)*float(y)
    sumxy += float(x)*float(y)
  return 1-(sumxy/math.sqrt(sumxx*sumyy))

def getdistances(data,vec1):
  distancelist=[]
```

```
46
47  K Nearest Neighbors for WS-DL:
48  When k = 1
49  .
50  When k = 2
51  .
52  La Fotografía Efectista Abstracta. Fotos Abstractas. Abstract  Photos.
53  When k = 5
54  .
55  La Fotografía Efectista Abstracta. Fotos Abstractas. Abstract  Photos.
56  Rosie Gigg A2 Media Studies
57  A Wife's Tale
58  Mile In Mine
59  When k = 10
60  .
61  La Fotografía Efectista Abstracta. Fotos Abstractas. Abstract  Photos.
62  Rosie Gigg A2 Media Studies
63  A Wife's Tale
64  Mile In Mine
65  Morgan's Blog
66  For the Other Things
67  IoTube     :)
68  Pithy Title Here
69  funky little demons
70  When k = 20
71  .
72  La Fotografía Efectista Abstracta. Fotos Abstractas. Abstract  Photos.
73  Rosie Gigg A2 Media Studies
74  A Wife's Tale
75  Mile In Mine
76  Morgan's Blog
77  For the Other Things
78  IoTube     :)
79  Pithy Title Here
80  funky little demons
81  My Name Is Blue Canary
82  sweet crimson blood
83  isyeli's
84  Stonehill Sketchbook
85  this time tomorrow
86  symmetry/symmetry
87  MTJR RANTS & RAVES ON MUSIC
88  theindiefriend
89  künstler treu
90  jaaackie.
91
```

```
16
17  # Loop over every item in the dataset
18  for i in range(len(data)):
19    vec2=data[i]
20    try:
21      distancelist.append((cosine(vec1,vec2),i))
22    except:
23      pass
24
25  # Sort by distance
26  distancelist.sort()
```

```
27   return distancelist
28
29 def knnestimate(data,vec1,k=5):
30   # Get sorted distances
31   dlist=getdistances(data,vec1)
32   avg=0.0
33   return dlist
```

Problem 2

Rerun A9, Q2 but this time using LIBSVM. If you have n categories, you'll have to run it n times. For example, if you're classifying music and have the categories:

metal, electronic, ambient, folk, hip-hop, pop

you'll have to classify things as:

metal / not-metal
electronic / not-electronic
ambient / not-ambient

etc.

Use the 500 term vectors describing each blog as the features, and your mannally assigned classifications as the true values. Use 10-fold cross-validation (as per slide 46, which shows 4-fold cross-validation) and report the percentage correct for each of your categories.

Answer

Problem 3

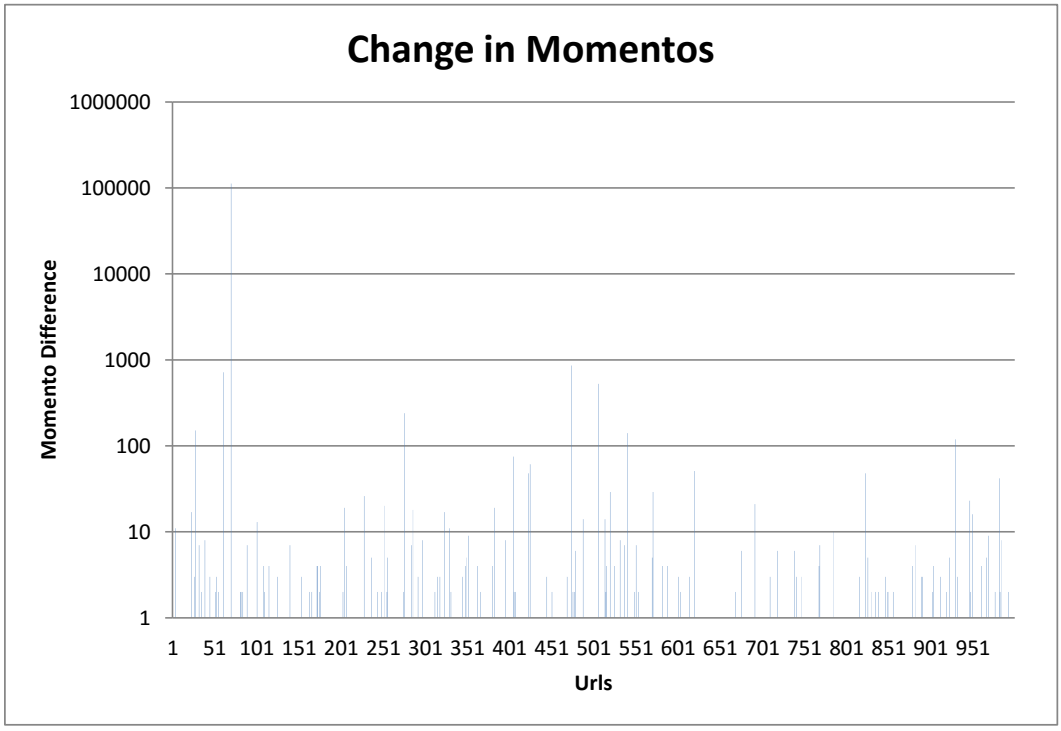
Re-download the 1000 TimeMaps from A2, Q2. Create a graph where the x-axis represents the 1000 TimeMaps. If a TimeMap has "shrunk", it will have a negative value below the x-axis corresponding to the size difference between the two TimeMaps. If it has stayed the same, it will have a "0" value. If it has grown, the value will be positive and correspond to the increase in size between the two

```
TimeMaps.

As always, upload all the TimeMap data.  If the A2 github has the
original TimeMaps, then you can just point to where they are in
the report.
```

Answer

To solve this problem, I downloaded the 1000 TimeMaps again and putted both the old memento count and new memento count into a spread sheet to do the calculation. To see the changes, I subtracted the old memento counts from the new memento counts. If the result of one subtraction is negative, then, this page has been removed from some archives. If the result is positive, then there are more archives of this web page. In my case, I do not have any negatives in my list of 1000 URIs. There are actually large amount of increment across the board. I used log scale in my y axis because of one URI that has very large amount of increment in memento counts than rest of them. All the TimeMap data are provided in folder Problem3 on my github account.



C2 | f(x) Σ = | =B2-A2

	A	B	C	D	E
1	old	new	difference		
2	0	0	0		
3	0	0	0		
4	0	0	0		
5	0	11	11		
6	0	0	0		
7	0	0	0		
8	0	0	0		
9	0	0	0		
10	0	0	0		
11	0	0	0		
12	0	0	0		
13	0	0	0		
14	1	2	1		
15	0	0	0		
16	0	0	0		
17	0	0	0		
18	0	0	0		
19	0	0	0		
20	0	0	0		
21	0	0	0		
22	0	0	0		
23	0	0	0		
24	6	23	17		

Figure 1: Sample of newsubold.xlsx

Problem 4

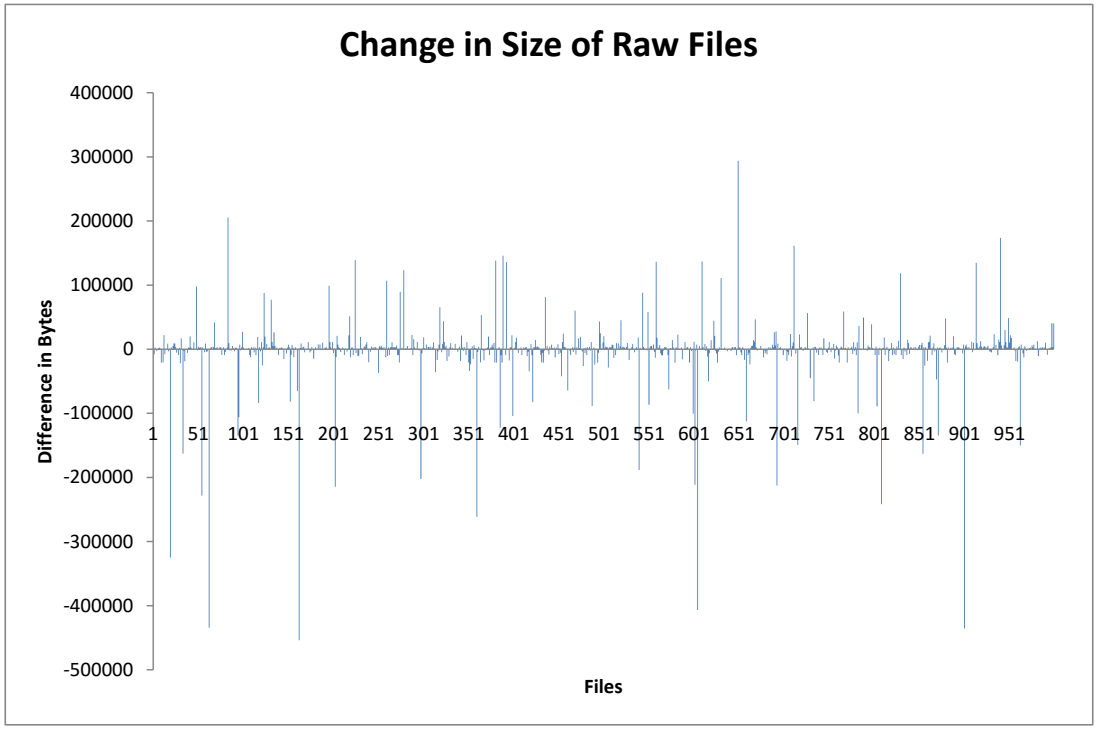
```
Repeat A3, Q1.  Compare the resulting text from February to
the text you have now.  Do all 1000 URIs still return a "200 OK"
as their final response (i.e., at the end of possible redirects)?

Create two graphs similar to that described in Q3, except this
time the y-axis corresponds to difference in bytes (and not difference
in TimeMap magnitudes).  For the first graph, use the difference
in the raw (unprocessed) results.  For the second graph, use the
difference in the processed (as per A3, Q1) results.
```

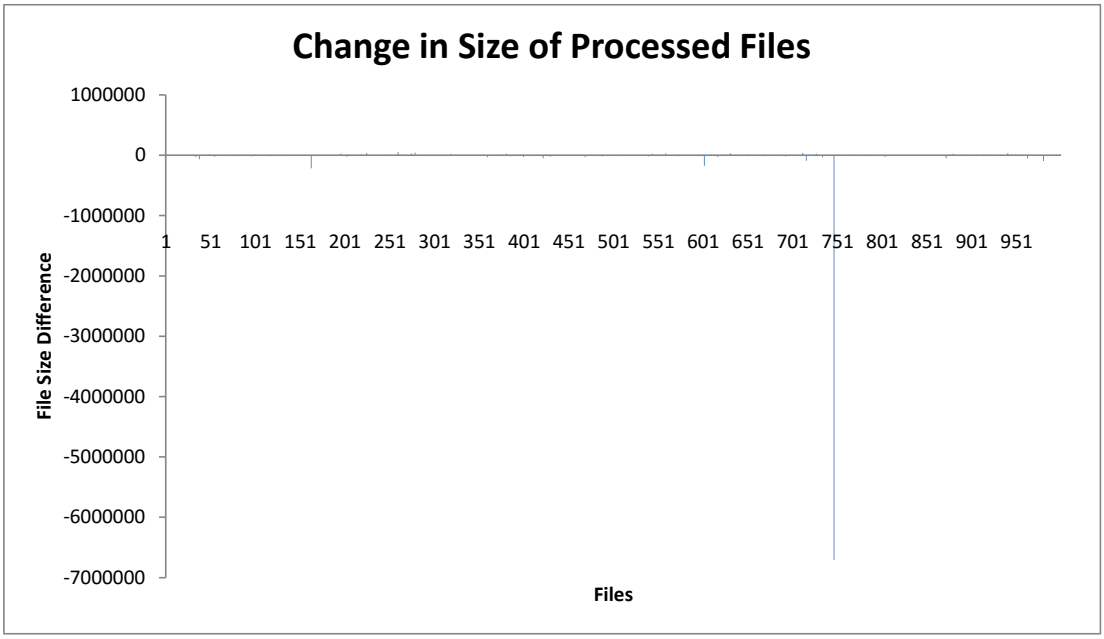
```
Of the URIs that still terminate in a "200 OK" response, pick the
top 3 most changed (processed) pairs of pages and use the Unix
"diff" command to explore the differences in the version pairs.
```

Answer

To solve this problem, I repeated the process in assignment 3 question 1. Then I wrote a piece of code to give me the file sizes of every old and new files. Then the results are put together in spread sheets to calculate the differences. All of the data and files are available in github.



In the processed files, between file 14.txt new and file 14.txt old, there is a huge amount of decrement in size comparing with other file size changes. Which made it hard to observe the graph accurately.




```
import os, sys

savefile = open('newProcessedSizes.txt', 'a')
path = "/home/hhuang/Documents/CS532/WebScienceAssig/Assignment10/Problem4/
    Processed/ProcessedFiles/"

for filename in os.listdir(path):
  filepath = os.path.join(path, filename)
  size = os.path.getsize(filepath)
  savefile.write(str(size))
  savefile.write('\n')
  print size
```

I use command "diff filename1 filename2" to get the differences of top 3 most changed pairs. The differences are saved in file names diff14.txt, diff276.txt, and diff571.txt. The files are too large to be shown in this report, they can be find in the "3most" folder in Problem4 folder on my github account.