2017

Assignment 10

CS 432

MIRANDA SMITH

Question 1

Using the data from A8:

- Consider each row in the blog-term matrix as a 1000-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 1000 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}.

I created the program 'usingknnEstimate.py' to parse the blog matrix and pass on the relevant data to knnestimate(). I used the 'clusters.py' program from Programming Collective Intelligence to parse the blog matrix. The indexes corresponding to the correct blogs were found manually: 0 for F-Measure and 77 for Web Science and Digital Libraries Research Group. knnEstimate() is then called to find the closest matches and print them out.

```
import numpredict
import clusters

blognames,words,data=clusters.readfile('blogdata1.txt')


#k={1,2,5,10,20}
print (blognames[0]+ " k =1")
numpredict.knnestimate(blognames[1:], data[1:],data[0], k=1)
print (blognames[0]+ " k =2")
numpredict.knnestimate(blognames[1:], data[1:],data[0], k=2)
print (blognames[0]+ " k =5")
numpredict.knnestimate(blognames[1:], data[1:],data[0], k=5)
print (blognames[0]+ " k =10")
numpredict.knnestimate(blognames[1:], data[1:],data[0], k=10)
print (blognames[0]+ " k =20")
numpredict.knnestimate(blognames[1:], data[1:],data[0], k=20)

blogname = blognames[77]
blogdata = data[77]
blognames.pop(77)
data.pop(77)
print (blogname + " k =1")
numpredict.knnestimate(blognames, data,blogdata, k=1)
print (blogname+ " k =2")
numpredict.knnestimate(blognames, data,blogdata, k=2)
print (blogname+ " k =5")
numpredict.knnestimate(blognames, data,blogdata, k=5)
print (blogname+ " k =10")
numpredict.knnestimate(blognames, data,blogdata, k=10)
print (blogname+ " k =20")
numpredict.knnestimate(blognames, data,blogdata, k=20)
```

I edited the following functions from ‘numpredict.py’ originally found in Programming Collective Intelligence.

```
def cosine(v1,v2):
  return spatial.distance.cosine(v1,v2)

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

  # Loop over every item in the dataset
  for i in range(len(data)):
    vec2=data[i]

    # Add the distance and the index
    distancelist.append((cosine(vec1,vec2),i))

  # Sort by distance
  distancelist.sort()
  return distancelist

def knnestimate(blognames, data,vec1,k=5):
  # Get sorted distances
  dlist=getdistances(data,vec1)


  # Take the average of the top k results
  for i in range(k):
    print(str(dlist[i][0]) + " - " + str(blognames[dlist[i][1]]) )


  return dlist
```

I created the cosine function to take advantage of the library scipy to find the cosine distance. I removed references to [‘index’] because the data is stored differently from the book examples. Knnestimate now takes another parameter, blognames, that corresponds to the names of the blogs in data. Instead of averaging the closest neighbors it just returns the list and print out each distance and the blog’s name.

Output:

```
F-Measure k =1
0.355801690142 - The World Through My Eyes
F-Measure k =2
0.355801690142 - The World Through My Eyes
0.363785656517 - Popping the Bubble
F-Measure k =5
0.355801690142 - The World Through My Eyes
0.363785656517 - Popping the Bubble
0.381594702807 - light your world
0.388952608182 - Misfit Hausfrau
0.397342996986 - Joining the Heavens
F-Measure k =10
0.355801690142 - The World Through My Eyes
0.363785656517 - Popping the Bubble
0.381594702807 - light your world
0.388952608182 - Misfit Hausfrau
0.397342996986 - Joining the Heavens
0.400513537627 - Peaceable Wisdom
0.40251056583 - Steve’s Soapbox
0.408231915813 - Nonfast Turtle Juice.
0.409644540343 - Texas Tabernacle by Jeff Clark
0.409797463769 - go, Go, GO!
F-Measure k =20
0.355801690142 - The World Through My Eyes
0.363785656517 - Popping the Bubble
0.381594702807 - light your world
0.388952608182 - Misfit Hausfrau
0.397342996986 - Joining the Heavens
0.400513537627 - Peaceable Wisdom
0.40251056583 - Steve’s Soapbox
0.408231915813 - Nonfast Turtle Juice.
0.409644540343 - Texas Tabernacle by Jeff Clark
0.409797463769 - go, Go, GO!
0.416671862597 - Gary @ Tamaki Primary School
0.418672534614 - The Art Of Going Mouldy
0.430526479372 - You Are the Sunshine of My Life
0.434373676971 - Redd Adventures
0.435955977024 - miles to go before i sleep...
0.43924218814 - Not So Feminist
0.441044649792 - CHRISTOPHER HAYDEN - Poetry
0.447906819259 - Stuff Of Dreams & Life
0.449127690136 - Bartlebee's Bumblings
0.45087220683 - sparkleandshine
```

```
Web Science and Digital Libraries Research Group k =1
0.443210054869 - My Notes
Web Science and Digital Libraries Research Group k =2
0.443210054869 - My Notes
0.458499031121 - hmmhannah
Web Science and Digital Libraries Research Group k =5
0.443210054869 - My Notes
0.458499031121 - hmmhannah
0.463823313434 - Iris Design Life
0.525986399954 - Opel Rekord A & B
0.531523551383 - Joining the Heavens
Web Science and Digital Libraries Research Group k =10
0.443210054869 - My Notes
0.458499031121 - hmmhannah
0.463823313434 - Iris Design Life
0.525986399954 - Opel Rekord A & B
0.531523551383 - Joining the Heavens
0.545279636882 - Really Useful Knowledge
0.547841359512 - Popping the Bubble
0.550952756969 - All Mehdiabad Students Organization Mehdiabad
0.553507436359 - Project Thesis NIU
0.564635302313 - More Randomness
Web Science and Digital Libraries Research Group k =20
0.443210054869 - My Notes
0.458499031121 - hmmhannah
0.463823313434 - Iris Design Life
0.525986399954 - Opel Rekord A & B
0.531523551383 - Joining the Heavens
0.545279636882 - Really Useful Knowledge
0.547841359512 - Popping the Bubble
0.550952756969 - All Mehdiabad Students Organization Mehdiabad
0.553507436359 - Project Thesis NIU
0.564635302313 - More Randomness
0.566895113884 - AbideInSelf - The Nonduality Blog
0.566966757498 - Modern Co.
0.58117496272 - DES32's BLOG
0.586847907964 - Gary @ Tamaki Primary School
0.589947316853 - My Not So Glamorous Life
0.590969885594 - Marbling
0.594820463359 - Texas Tabernacle by Jeff Clark
0.599891871273 - 3Coins
0.602282879603 - Not So Feminist
0.603022125022 - pieces of me...
```

Question 4

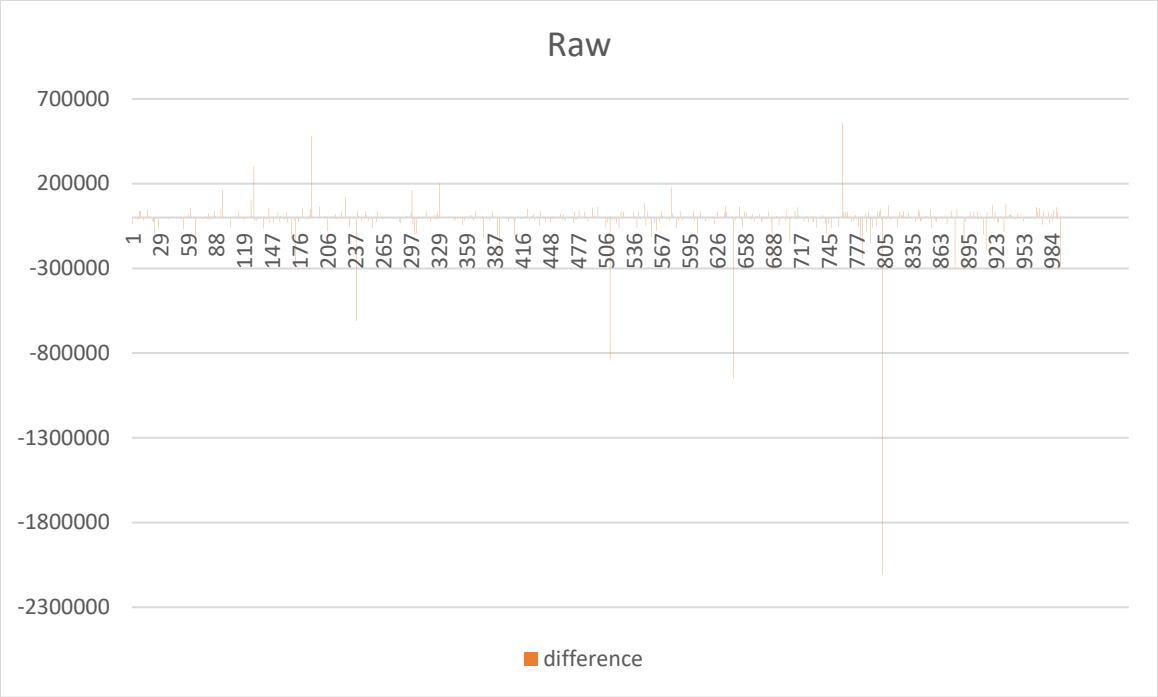
Repeat A3, Q1. Compare the resulting text from February to the text you have now. Do all 1000 URIs still return a "200 OK" as their final response (i.e., at the end of possible redirects)?

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.

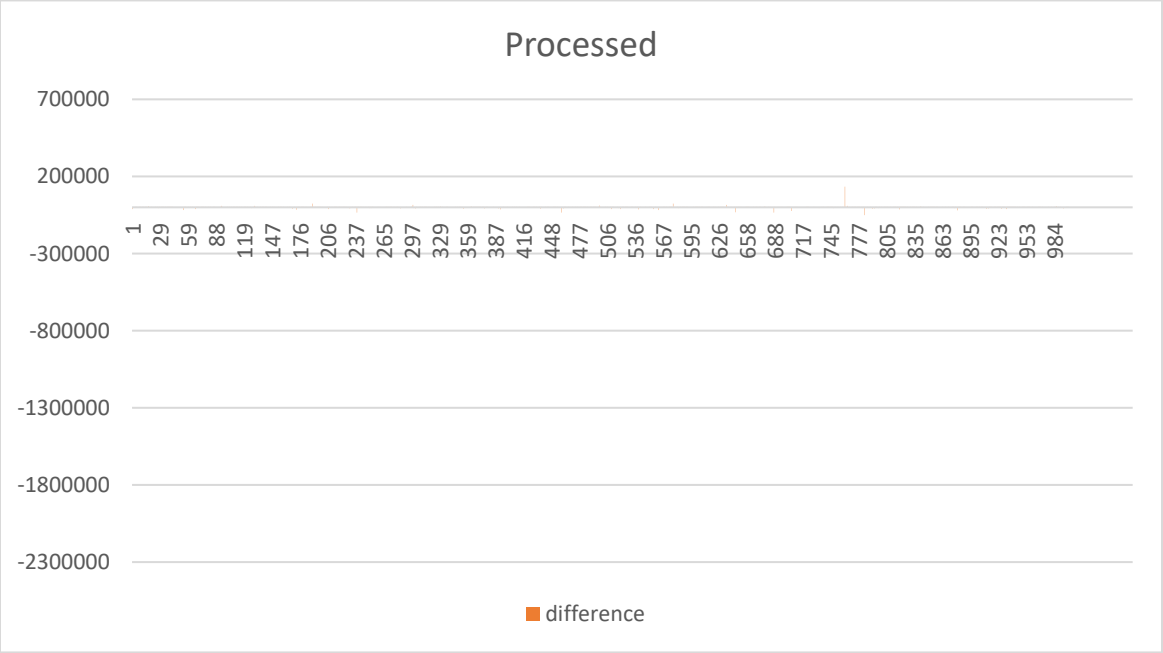
Of the URIs that still terminate in a "200 OK" response, pick the top 3 most changed (processed) pairs of pages and use the Unix "diff" command to explore the differences in the version pairs.

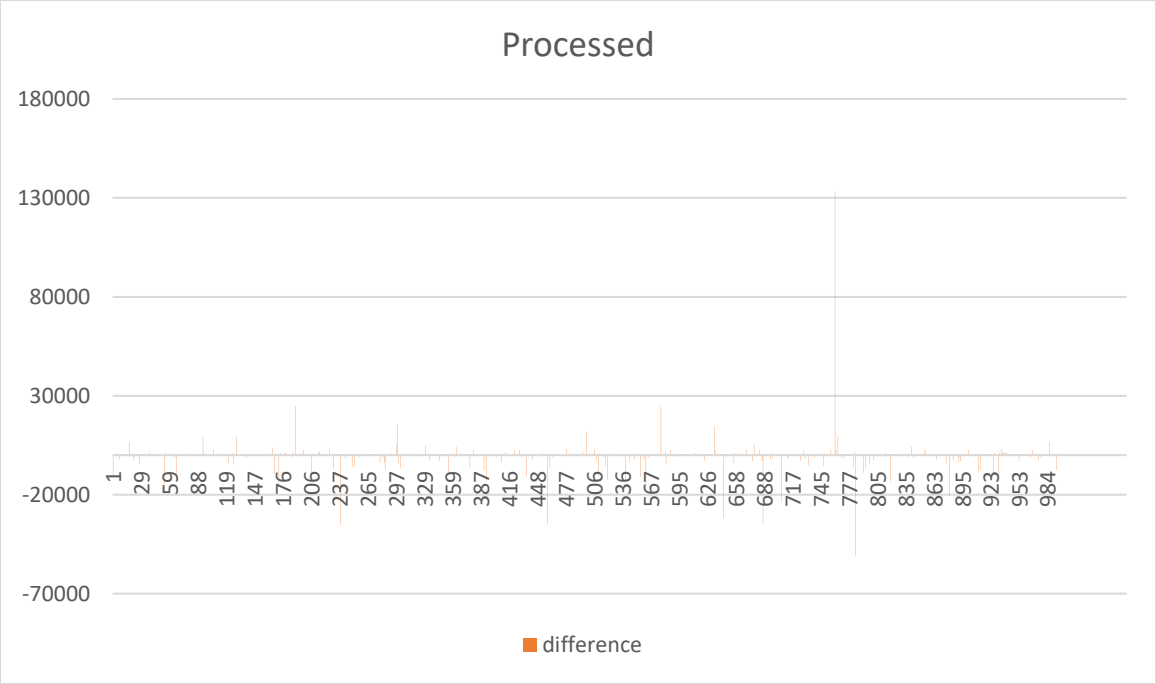
I used the same 'gatherHTML' and 'processHTML' that I used for A3 to collect the text. I created a small python program 'getFileSize.py' to grab the file sizes and write it to a text file so I could put the data into excel.

There were 29 URIs that did not return code 200.



With this scale on the processed, you can hardly see the nuances of the graph. However, it is important to show the two graphs at the same scale to appreciate the much greater changes the Raw files have undergone. Showing the formatting and behind the scenes changes much more than the content has.





A zoomed in view.

I used diff, with suppressed commonalities and in column form for each of the top 3 most changed. They are in differences189, differences578, and differences762.

They are too large to place in the report. However, most of the changes seem to be in menus and there is a lot of white space. The only one that looks like it had some content change is 762 and it's in another language so I can't be sure.