

Assignment 8

CS 532: Introduction to Web Science

Spring 2017

Grant Atkins

Finished on April 13, 2017

1

Question

1. Create a blog-term matrix. Start by grabbing 100 blogs; include:

<http://f-measure.blogspot.com/>

<http://ws-dl.blogspot.com/>

and grab 98 more as per the method shown in class. Note that this method randomly chooses blogs and each student will separately do this process, so it is unlikely that these 98 blogs will be shared among students. In other words, no sharing of blog data. Upload to github your code for grabbing the blogs and provide a list of blog URIs, both in the report and in github.

Use the blog title as the identifier for each blog (and row of the matrix). Use the terms from every item/title (RSS) or entry/title (Atom) for the columns of the matrix. The values are the frequency of occurrence. Essentially you are replicating the format of the "blogdata.txt" file included with the PCI book code. Limit the number of terms to the most "popular" (i.e., frequent) 1000 terms, this is *after* the criteria on p. 32 (slide 7) has been satisfied. Remember that blogs are paginated.

Answer

To solve this problem I decided to write a shell script called **grabBlogs.sh** to retrieve the 98 random blogs as shown in Listing 1. The script utilizes mainly the curl and sort commands to solve the problem of retrieving unique blogs. The curl command is called 200 times with `http://www.blogger.com/next-blog?navBar=true&blogID=3471633091411211117` called each time to retrieve 200 blogs. I used 200 executions because I happened to get duplicate blogs on some occasions and 100 calls wouldn't be enough to satisfy the requirements of this problem. For each new blog I got I saved the contents of the html page to a html document with the an id of the current iteration in the range of 200. I saved this file id and the URI found inside a file called **blogList.txt** as shown in Listing 2.

After 200 blogs were retrieved I used the sort command to find all unique blogs in the blog list file. Once this completed I had approximately 120 unique blogs. I then wrote a script in python 3.6 called **getFeed.py** as shown in Listing 3 which parsed each html document saved using the library Beautiful soup which allowed me to search the document by an HTML 'link' element to find the atom+xml feed of the blog [2]. I saved these feeds to a file called **feedList.txt** which was later cleaned for any blogs that did not have atom+xml feeds. Usually these were blogs that no longer existed. Any extra blogs above 100 were simply discarded.

Finally I proceeded to to write another python script called **generateFeedVector.py** shown in Listing 4 which utilized the code provide by the Programming Collective Intelligence (PCI) book [3]. This script was slightly modified to be usable for python 3.6 but also adding a limit to the amount of words the blog-term matrix allowed to a maximum of 1000 shown on line 65. I also didn't check for stop words when I retrieved the atom feeds, but I did check if words were stop words before creating the the matrix by using the nltk library's corpus of stop words shown on line 28. If a word was found to be a stop word according to their corpus it would not be added to the matrix. When this was completed it was saved to a file called **blogData.txt**.

```
1 #!/bin/bash
2
3 # list of blogs
4 blogList=data/blogList.txt
5 if [ -f "$blogList" ]; then
6     rm -f $blogList
7 fi
8
9 touch $blogList
10
```

```

11 # downloaded html blogs
12 blogDir=data/blogs/
13 if [ -d "$blogDir" ]; then
14     rm -r $blogDir
15 fi
16
17 mkdir $blogDir
18
19 curl -s "http://f-measure.blogspot.com/" > "${blogDir}b001.html"
20 curl -s "http://ws-dl.blogspot.com/" > "${blogDir}b002.html"
21
22 echo "b001.html http://f-measure.blogspot.com/" >> $blogList
23 echo "b002.html http://ws-dl.blogspot.com/" >> $blogList
24
25 for (( i = 3; i <= 200; i++ )); do
26     num='seq -f%03g $i $i'
27     uri='curl -Ls -o data/b$num.html -w %{url_effective} "
28         http://www.blogger.com/next-blog?navBar=true&blogID
29         =3471633091411211117"'
30     echo "b$num.html $uri" >> $blogList
31 done
32 #remove duplicate uri and page files
33 sort -u -k2 $blogList > data/tempList
34 sort -k1 data/tempList > $blogList
35 rm data/tempList
36
37 for file in `cat $blogList | cut -d' ' -f1`; do
38     mv data/$file $blogDir
39 done
40
41 # cleanup duplicate downloaded files
42 toDelete=$(find ./data -maxdepth 1 -name "*b*.html")
43 for item in $toDelete
44 do
45     rm $item
46 done

```

Listing 1: Shell script to retrieve unique blogs

```

1 b001.html http://f-measure.blogspot.com/
2 b002.html http://ws-dl.blogspot.com/
3 b003.html http://macthemost.blogspot.com/?expref=next-blog
4 b004.html http://itll-glow-on-you.blogspot.com/?expref=next-blog
5 b005.html http://themusicbinge.blogspot.com/?expref=next-blog
6 b006.html http://tmcbyrone.blogspot.com/?expref=next-blog
7 b007.html http://mccookerybook.blogspot.com/?expref=next-blog
8 b008.html http://revolverusa.blogspot.com/?expref=next-blog

```

9	b009.html	http://ahtapotunbahcesi.blogspot.com/?expref=next-blog
10	b010.html	http://cherryarea.blogspot.com/?expref=next-blog
11	b011.html	http://indietop20.blogspot.com/?expref=next-blog
12	b012.html	http://stonehillsketchbook.blogspot.com/?expref=next-blog
13	b013.html	http://cuzmusicrocks.blogspot.com/?expref=next-blog
14	b014.html	http://glipress.blogspot.com/?expref=next-blog
15	b016.html	http://truthfulmood.blogspot.com/?expref=next-blog
16	b017.html	http://www.sunstockmusic.com/?expref=next-blog
17	b018.html	http://markfishers-musicreview.blogspot.com/?expref=next-blog
18	b019.html	http://ohyesjonsi.blogspot.com/?expref=next-blog
19	b020.html	http://blog.spinitron.com/?expref=next-blog
20	b021.html	http://lost-places-hamburg.blogspot.com/?expref=next-blog
21	b022.html	http://antonellagiugliano.blogspot.com/?expref=next-blog
22	b023.html	http://storiesfromthecityradiovalencia.blogspot.com/?expref=next-blog
23	b024.html	http://onwarmermusic.blogspot.com/?expref=next-blog
24	b025.html	http://floorshimezipperboots.blogspot.com/?expref=next-blog
25	b026.html	http://noradiorecs.blogspot.com/?expref=next-blog
26	b027.html	http://www.chrisanne-grise.com/?expref=next-blog
27	b029.html	http://momentarilymusical.blogspot.com/?expref=next-blog
28	b030.html	http://maggotcaviar.blogspot.com/?expref=next-blog
29	b032.html	http://stephanieveto.blogspot.com/?expref=next-blog
30	b035.html	http://mondaywakeup.blogspot.com/?expref=next-blog
31	b036.html	http://www.mpcfilmco.com/?expref=next-blog
32	b037.html	http://justwordsnomeaning.blogspot.com/?expref=next-blog
33	b038.html	http://blog.spin-itrecords.ca/?expref=next-blog
34	b039.html	http://davecromwellwrites.blogspot.com/?expref=next-blog
35	b040.html	http://mobbie2.blogspot.com/?expref=next-blog
36	b041.html	http://intheframefilmreviews.blogspot.com/?expref=next-blog
37	b042.html	http://mileinmine.blogspot.com/?expref=next-blog
38	b043.html	http://insearchoftrueromantics.blogspot.com/?expref=next-blog
39	b044.html	http://bonjourgirl.blogspot.com/?expref=next-blog
40	b045.html	http://didnotchart.blogspot.com/?expref=next-blog
41	b046.html	http://wildnightstranger.blogspot.com/?expref=next-blog
42	b047.html	http://mesastivromia.blogspot.com/?expref=next-blog
43	b048.html	http://bartkings.blogspot.com/?expref=next-blog
44	b049.html	http://jlmddlcm1516.blogspot.com/?expref=next-blog
45	b053.html	http://abueveryday.blogspot.com/?expref=next-blog

46	b054.html	http://doyouneedatv.blogspot.com/?expref=next-blog
47	b055.html	http://srkikiblog.blogspot.com/?expref=next-blog
48	b056.html	http://bleakbliss.blogspot.com/?expref=next-blog
49	b058.html	http://dancingincirclesnow.blogspot.com/?expref=next-blog
50	b059.html	http://ridingaborrowedbike.blogspot.com/?expref=next-blog
51	b060.html	http://www.hipindetroit.com/?expref=next-blog
52	b061.html	http://marshwiggle.blogspot.com/?expref=next-blog
53	b063.html	http://sixeyes.blogspot.com/?expref=next-blog
54	b064.html	http://encorenorthernireland.blogspot.com/?expref=next-blog
55	b065.html	http://johnandmaureensanto.blogspot.com/?expref=next-blog
56	b068.html	http://franbrighton.blogspot.com/?expref=next-blog
57	b069.html	http://mtjrrantsravesonmusic.blogspot.com/?expref=next-blog
58	b071.html	http://turnitupjack.blogspot.com/?expref=next-blog
59	b073.html	http://outanddowninthecolonies.blogspot.com/?expref=next-blog
60	b074.html	http://organmyth.blogspot.com/?expref=next-blog
61	b080.html	http://jojobethkatiehannahlcm1516.blogspot.com/?expref=next-blog
62	b081.html	http://tuquesveristo.blogspot.com/?expref=next-blog
63	b082.html	http://jbreitling.blogspot.com/?expref=next-blog
64	b083.html	http://paulinag-mediaa2.blogspot.com/?expref=next-blog
65	b084.html	http://www.thejeopardyofcontentment.com/?expref=next-blog
66	b085.html	http://www.juanbook.com/?expref=next-blog
67	b086.html	http://skinnyshoes.blogspot.com/?expref=next-blog
68	b087.html	http://steel-city-rust.blogspot.com/?expref=next-blog
69	b090.html	http://fridaynightdream.blogspot.com/?expref=next-blog
70	b094.html	http://avidsblog.blogspot.com/?expref=next-blog
71	b096.html	http://atozappa.blogspot.com/?expref=next-blog
72	b099.html	http://ps-music.blogspot.com/?expref=next-blog
73	b105.html	http://primitiveofferings.blogspot.com/?expref=next-blog
74	b107.html	http://markeortega.blogspot.com/?expref=next-blog
75	b110.html	http://londynsky.blogspot.com/?expref=next-blog
76	b111.html	http://somecallitnoise.blogspot.com/?expref=next-blog
77	b112.html	http://liquid-pop.blogspot.com/?expref=next-blog
78	b113.html	http://elijace.blogspot.com/?expref=next-blog
79	b116.html	http://theworldsfirstinternetbaby.blogspot.com/?expref=next-blog
80	b117.html	http://dazeyrosie.blogspot.com/?expref=next-blog
81	b118.html	http://earenjoy.blogspot.com/?expref=next-blog
82	b120.html	http://my-name-is-blue-canary.blogspot.com/?expref=next-blog
83	b124.html	http://www.thestarkonline.com/?expref=next-blog

```

84 b131.html http://dana9morgan.blogspot.com/?expref=next-blog
85 b132.html http://hellomynameisjustin.blogspot.com/?expref=next-
    blog
86 b139.html http://paradoxical-era.blogspot.com/?expref=next-blog
87 b141.html http://pithytittlehere.blogspot.com/?expref=next-blog
88 b145.html http://myopiamuse.blogspot.com/?expref=next-blog
89 b147.html http://jamiemclelland.blogspot.com/?expref=next-blog
90 b150.html http://ngaio1619.blogspot.com/?expref=next-blog
91 b151.html http://aubadel.blogspot.com/?expref=next-blog
92 b152.html http://hani-bittersweet.blogspot.com/?expref=next-blog
93 b153.html http://worldofpearljambootlegs.blogspot.com/?expref=
    next-blog
94 b156.html http://isyelili.blogspot.com/?expref=next-blog
95 b158.html http://adrianomarquesblog.blogspot.com/?expref=next-
    blog
96 b161.html http://naoponhomusica.blogspot.com/?expref=next-blog
97 b165.html http://dcresider.blogspot.com/?expref=next-blog
98 b169.html http://fractalpress.blogspot.com/?expref=next-blog
99 b171.html http://doginasweatershowreviews.blogspot.com/?expref=
    next-blog
100 b173.html http://flowradio.blogspot.com/?expref=next-blog

```

Listing 2: 100 unique blogs collected

```

1
2 import requests
3 from bs4 import BeautifulSoup
4 import os
5
6
7 def getFeed(filename):
8     print("FILENAME:", filename)
9     with open("data/blogs/" + filename) as f:
10         f.seek(0)
11         html = f.read()
12         soup = BeautifulSoup(html, 'html.parser')
13         feed = soup.find_all(
14             'link', attrs={'type': 'application/atom+xml'})
15
16         if (feed):
17             return feed[0]['href']
18         return None
19
20
21 def deleteNoFeedFiles():
22     noneLines = []
23     f = open("data/feedList.txt", "r+")
24     d = f.readlines()
25     f.seek(0)

```

```

26     for i, line in enumerate(d):
27         if 'None' in line:
28             noneLines.append(i)
29         else:
30             f.write(line)
31     f.truncate()
32     f.close()
33
34     print(noneLines)
35     f = open("data/blogList.txt", "r+")
36     d = f.readlines()
37     f.seek(0)
38     for i, line in enumerate(d):
39         if i in noneLines:
40             # delete file
41             fileToDelete = line.split(' ')[0]
42             print("Deleting:", fileToDelete)
43             os.remove("data/blogs/" + fileToDelete)
44         else:
45             f.write(line)
46     f.truncate()
47     f.close()
48
49
50 if __name__ == "__main__":
51
52     with open("data/blogList.txt") as f, open("data/feedList.txt",
53         "w") as out:
54         for line in f:
55             filename = line.split(' ')[0]
56             feed = getFeed(filename)
57             print(feed, file=out)
58
59     deleteNoFeedFiles()

```

Listing 3: Python script to retrieve atom feeds of blogs

```

1  #!/usr/bin/python
2  # -*- coding: utf-8 -*-
3  import feedparser
4  import re
5  from nltk.corpus import stopwords
6
7  stops = stopwords.words("english")
8
9  def getwordcounts(url):
10     '''
11     Returns title and dictionary of word counts for an RSS feed
12     '''

```



```

13     # Parse the feed
14     d = feedparser.parse(url)
15     wc = {}
16
17     # Loop over all the entries
18     for e in d.entries:
19         if 'summary' in e:
20             summary = e.summary
21
22         else:
23             summary = e.description
24
25         # Extract a list of words
26         words = getwords(e.title + ' ' + summary)
27         for word in words:
28             if word not in stops:
29                 wc.setdefault(word, 0)
30                 wc[word] += 1
31
32     return (d.feed.title, wc)
33
34
35 def getwords(html):
36     # Remove all the HTML tags
37     txt = re.compile(r'<[^>]+>').sub('', html)
38
39     # Split words by all non-alpha characters
40     words = re.compile(r'[^A-Za-z]+').split(txt)
41
42     # Convert to lowercase
43     return [word.lower() for word in words if word != '']
44
45
46 apcount = {}
47 wordcounts = {}
48 feedlist = [line for line in open('data/feedList.txt')]
49 for feedurl in feedlist:
50     try:
51         (title, wc) = getwordcounts(feedurl)
52         wordcounts[title] = wc
53         for (word, count) in wc.items():
54             apcount.setdefault(word, 0)
55             if count > 1:
56                 apcount[word] += 1
57     except:
58         print('Failed to parse feed', feedurl)
59
60 wordlist = []
61 for (w, bc) in apcount.items():

```

```

62     frac = float(bc) / len(feedlist)
63     if frac > 0.1 and frac < 0.5:
64         wordlist.append(w)
65     if len(wordlist) >= 1000:
66         break
67
68 out = open('data/blogdata.txt', 'w')
69 out.write('Blog')
70 for word in wordlist:
71     out.write('\t%s' % word)
72 out.write('\n')
73 for (blog, wc) in wordcounts.items():
74     print(blog)
75     out.write(blog)
76     for word in wordlist:
77         if word in wc:
78             out.write('\t%d' % wc[word])
79         else:
80             out.write('\t0')
81 out.write('\n')

```

Listing 4: Python script to generate blog-term matrix

2

Question

2. Create an ASCII and JPEG dendrogram that clusters (i.e., HAC) the most similar blogs (see slides 12 & 13). Include the JPEG in your report and upload the ascii file to github (it will be too unwieldy for inclusion in the report).

Answer

To solve this question I used the code provided by the Programming Collective Intelligence book to write a script in python 2.7 called **createClusters.py** as shown in Listing 5 [3]. This script has a method called *createDendrogram* which utilizes the clusters.py file provided by the PCI book load the blog-term matrix created in question 1 to create a Hierarchical Clustering tree image, the dendrogram, as shown in Figure 1. I also created an ASCII file named **q2ASCII.txt** to represent this tree structure in text which is available on my Github page [1].

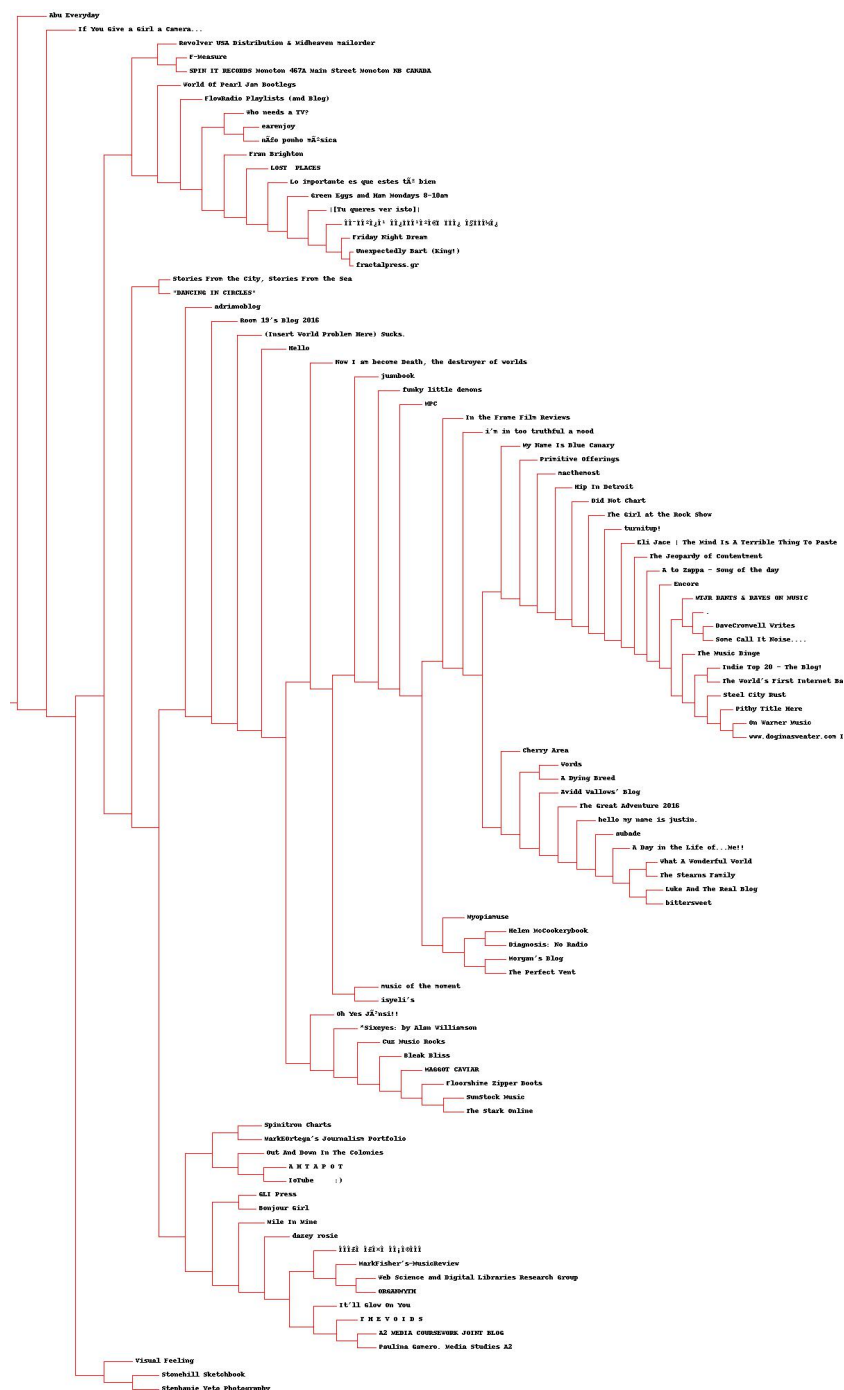


Figure 1: Dendrogram for blogs collected

```

1
2 import clusters
3 import sys
4
5
6 def createDendrogram():
7     blogs, colnames, data = clusters.readfile('data/blogdata.txt
8         ')
9     cluster = clusters.hcluster(data)
10    clusters.drawdendrogram(cluster, blogs, jpeg='../docs/
11        q2Dendrogram.jpg')
12    f = open("data/q2ASCII.txt", 'w')
13    sys.stdout = f
14    clusters.printclust(cluster, labels=blogs)
15    f.close()
16    sys.stderr.close()
17
18 def kmeans():
19     karr = [5, 10, 20]
20     blogs, colnames, data = clusters.readfile('data/blogdata.txt
21         ')
22     for i in karr:
23         kclust, itercount = clusters.kcluster(data, k=i)
24         print(kclust)
25         f = open("data/kclust_%d.txt" % i, 'w')
26         f.write("Iteration count: %d \n" % itercount)
27         print(len(kclust))
28         for cluster in kclust:
29             f.write("*****\n")
30             f.write("[")
31             for blogid in cluster:
32                 f.write(blogs[blogid] + ", ")
33             f.write("]\n")
34
35 def mds():
36     blognames, words, data = clusters.readfile('data/blogdata.
37         txt')
38     coords, itercount = clusters.scaledown(data)
39     clusters.draw2d(coords, labels=blognames, jpeg='../docs/
40         q4Mds.jpg')
41     print('Iteration count: %d' % itercount)
42
43 if __name__ == "__main__":
44     ,,,

```

```
44     Written in python 2.7
45     '''
46     createDendrogram()
47     kmeans()
48     mds()
```

Listing 5: Python script to generate clusters with different approaches

3

Question

3. Cluster the blogs using K-Means, using $k=5,10,20$. (see slide 18). Print the values in each centroid, for each value of k . How many iterations were required for each value of k ?

Answer

To solve this question I again used the code provided by the Programming Collective Intelligence book as shown in Listing 5 to create a method called *kmeans* [3]. The *kmeans* method again reads my blog-term matrix and using the method *kcluster* for the clusters.py library. I modified the *kcluster* method to also return the iteration count so it could be saved along with the blog title. For each value of k I created a separate file named **kclut_n.txt** where n is the value of k . For a k value of 5 it took 7 iterations with the values of each centroids shown in Listing 6. For a k value of 10 it took 5 iterations with the values of each centroids shown in Listing 7. For a k value of 20 it took 5 iterations with the values of each centroids shown in Listing 8.

```
1 Iteration count: 7
2 *****
3 [Web Science and Digital Libraries Research Group, machthemost,
  The Music Binge, Indie Top 20 – The Blog!, Cuz Music Rocks, i
  'm in too truthful a mood, SunStock Music, MarkFisher's–
  MusicReview, Oh Yes J nsi!!, On Warmer Music, Floorshine
  Zipper Boots, The Girl at the Rock Show, MAGGOT CAVIAR,
  DaveCromwell Writes, In the Frame Film Reviews, Mile In Mine,
  Bonjour Girl, Did Not Chart, Bleak Bliss, Hip In Detroit, *
  Sixeyes: by Alan Williamson, Encore, MTJR RANTS & RAVES ON
  MUSIC, turnitup!, ., The Jeopardy of Contentment, Steel City
  Rust, A to Zappa – Song of the day, Primitive Offerings,
  MarkeOrtega's Journalism Portfolio, Some Call It Noise....,
  Eli Jace | The Mind Is A Terrible Thing To Paste, The World's
  First Internet Baby, The Stark Online, www.doginasweater.com
  Live Show Review Archive, ]
4 *****
5 [A H T A P O T, Spinitron Charts, LOST PLACES, Stories From the
  City, Stories From the Sea, Green Eggs and Ham Mondays 8–10
  am, , Unexpectedly Bart (King!),
  A2 MEDIA COURSEWORK JOINT BLOG, Lo importante es que estes
  t bien, "DANCING IN CIRCLES", Out And Down In The Colonies,
  ORGANMYTH, T H E V O I D S, |[Tu quieres ver isto]|, Paulina
  Gamero. Media Studies A2, If You Give a Girl a Camera...,
```

```

Friday Night Dream,
, My Name Is Blue Canary, fractalpress.gr, ]
6 *****
7 [F-Measure, Revolver USA Distribution & Midheaven mailorder,
SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA,
Abu Everyday, ]
8 *****
9 [It 'll Glow On You, Stephanie Veto Photography, IoTube :),
Who needs a TV?, Fran Brighton, dazey rosie, earenjoy, World
Of Pearl Jam Bootlegs, adrianoblog, n o ponho m sica,
FlowRadio Playlists (and Blog), ]
10 *****
11 [(Insert World Problem Here) Sucks., Helen McCookerybook, Cherry
Area, Stonehill Sketchbook, GLI Press, Visual Feeling,
Diagnosis: No Radio, music of the moment, MPC, Words, A Dying
Breed, Hello, Luke And The Real Blog, funky little demons,
The Great Adventure 2016, juanbook, Avidd Wallows' Blog, What
A Wonderful World, Now I am become Death, the destroyer of
worlds, Morgan's Blog, hello my name is justin., The Perfect
Vent, Pithy Title Here, Myopiamuse, The Stearns Family, Room
19's Blog 2016, aubade, bittersweet, isyeli's, A Day in the
Life of...Me!!, ]

```

Listing 6: K-means clustering with a value of 5

```

1 Iteration count: 5
2 *****
3 [SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA, ]
4 *****
5 [(Insert World Problem Here) Sucks., Helen McCookerybook, A H T
A P O T, Cherry Area, i'm in too truthful a mood, LOST
PLACES, Diagnosis: No Radio, music of the moment, Green Eggs
and Ham Mondays 8-10am, IoTube :), Bonjour Girl, Lo
importante es que estes t bien, Luke And The Real Blog,
funky little demons, Fran Brighton, Avidd Wallows' Blog,
Morgan's Blog, bittersweet, isyeli's, FlowRadio Playlists (
and Blog), ]
6 *****
7 [Bleak Bliss, ]
8 *****
9 [Stonehill Sketchbook, Stories From the City, Stories From the
Sea, Stephanie Veto Photography, Hello, "DANCING IN CIRCLES",
The Great Adventure 2016, If You Give a Girl a Camera...,
What A Wonderful World, The Stearns Family, Room 19's Blog
2016, World Of Pearl Jam Bootlegs, adrianoblog, n o ponho
m sica, ]
10 *****
11 [macthemost, The Music Binge, Revolver USA Distribution &
Midheaven mailorder, Cuz Music Rocks, SunStock Music,

```



```

MarkFisher's-MusicReview, Oh Yes J nsi!!, Floorshine Zipper
Boots, MAGGOT CAVIAR, DaveCromwell Writes, Did Not Chart, *
Sixeyes: by Alan Williamson, Encore, MTJR RANTS & RAVES ON
MUSIC, turnitup!, ., The Jeopardy of Contentment, Some Call
It Noise...., Eli Jace | The Mind Is A Terrible Thing To
Paste, The Stark Online, ]
12 *****
13 [Words, A Dying Breed, juanbook, Now I am become Death, the
destroyer of worlds, hello my name is justin., The Perfect
Vent, A Day in the Life of...Me!!, ]
14 *****
15 [Web Science and Digital Libraries Research Group, GLI Press,
, ORGANMYTH, dazey rosie ,
earenjoy, ]
16 *****
17 [It 'll Glow On You, Spinitron Charts, Unexpectedly Bart (King!),
A2 MEDIA COURSEWORK JOINT BLOG, Who needs a TV?, Out And
Down In The Colonies, |[Tu quieres ver isto]|, Paulina Gamero.
Media Studies A2, Friday Night Dream,
, fractalpress.gr, ]
18 *****
19 [F-Measure, Abu Everyday, T H E V O I D S, ]
20 *****
21 [Indie Top 20 - The Blog!, Visual Feeling, On Warmer Music, The
Girl at the Rock Show, MPC, In the Frame Film Reviews, Mile
In Mine, Hip In Detroit, Steel City Rust, A to Zappa - Song
of the day, Primitive Offerings, MarkeOrtega's Journalism
Portfolio, The World's First Internet Baby, My Name Is Blue
Canary, Pithy Title Here, Myopiamuse, aubade, www.
doginasweater.com Live Show Review Archive, ]

```

Listing 7: K-means clustering with a value of 10

```

1 Iteration count: 5
2 *****
3 [(Insert World Problem Here) Sucks., Helen McCookerybook, Cherry
Area, Stonehill Sketchbook, Diagnosis: No Radio, music of
the moment, Words, A Dying Breed, Hello, Luke And The Real
Blog, funky little demons, The Great Adventure 2016, juanbook
, Avidd Wallows' Blog, What A Wonderful World, Now I am
become Death, the destroyer of worlds, Morgan's Blog, hello
my name is justin., The Perfect Vent, The Stearns Family,
Room 19's Blog 2016, aubade, bittersweet, isyeli's, A Day in
the Life of...Me!!, ]
4 *****
5 []
6 *****
7 [Revolver USA Distribution & Midheaven mailorder, T H E V O I D
S, ]

```

```

8 *****
9 [SPIN IT RECORDS Moncton 467A Main Street Moncton NB CANADA, ]
10 *****
11 []
12 *****
13 [It 'll Glow On You, GLI Press, Bonjour Girl, Bleak Bliss,
    Paulina Gamero. Media Studies A2, The Jeopardy of Contentment
    , World Of Pearl Jam Bootlegs, ]
14 *****
15 []
16 *****
17 [F-Measure, ]
18 *****
19 []
20 *****
21 [Who needs a TV?, If You Give a Girl a Camera..., earenjoy, n o
    ponho m sica, FlowRadio Playlists (and Blog), ]
22 *****
23 [Oh Yes J nsi!!, Visual Feeling, DaveCromwell Writes, A2 MEDIA
    COURSEWORK JOINT BLOG, *Sixeyes: by Alan Williamson, The
    World's First Internet Baby, ]
24 *****
25 [Spinitron Charts, LOST PLACES, Green Eggs and Ham Mondays 8-10
    am, IoTube :), Unexpectedly Bart (King!), Lo importante
    es que estes t bien, Out And Down In The Colonies, |[Tu
    queres ver isto]|, Friday Night Dream,
    , adrianoblog, fractalpress
    .gr, ]
26 *****
27 [MarkFisher's-MusicReview, Stories From the City, Stories From
    the Sea, "DANCING IN CIRCLES", ]
28 *****
29 [Abu Everyday, ]
30 *****
31 [The Music Binge, Indie Top 20 - The Blog!, Cuz Music Rocks, i'm
    in too truthful a mood, SunStock Music, On Warmer Music,
    Floorshime Zipper Boots, The Girl at the Rock Show, MAGGOT
    CAVIAR, Stephanie Veto Photography, MPC, Mile In Mine, Encore
    , MTJR RANTS & RAVES ON MUSIC, turnitup!, Steel City Rust, A
    to Zappa - Song of the day, Eli Jace | The Mind Is A Terrible
    Thing To Paste, The Stark Online, Pithy Title Here,
    Myopiamuse, ]
32 *****
33 [Web Science and Digital Libraries Research Group,
    , ORGANMYTH, dazey rosie, ]
34 *****
35 [A H T A P O T, In the Frame Film Reviews, ]
36 *****
37 []

```

```

38 *****
39 [macthemost, Did Not Chart, Hip In Detroit, ., Primitive
    Offerings, MarkEOrtega's Journalism Portfolio, Some Call It
    Noise...., My Name Is Blue Canary, www.doginasweater.com Live
    Show Review Archive, ]
40 *****
41 [Fran Brighton, ]

```

Listing 8: K-means clustering with a value of 20

4

Question

4. Use MDS to create a JPEG of the blogs similar to slide 29 of the week 12 lecture. How many iterations were required?

Answer

5

Question

5. Re-run question 2, but this time with proper TFIDF calculations instead of the hack discussed on slide 7 (p. 32). Use the same 1000 words, but this time replace their frequency count with TFIDF scores as computed in assignment #3. Document the code, techniques, methods, etc. used to generate these TFIDF values. Upload the new data file to github.

Compare and contrast the resulting dendrogram with the dendrogram from question #2.

Note: ideally you would not reuse the same 1000 terms and instead come up with TFIDF scores for all the terms and then choose the top 1000 from that list, but I'm trying to limit the amount of work necessary.

Answer

NOT ATTEMPTED

6

Question

6. Re-run questions 1-4, but this time instead of using the 98 "random" blogs, use 98 blogs that should be "similar" to:

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

Choose approximately equal numbers for both blog sets (it doesn't have to be a perfect 49-49 split, but it should be close). Explain in detail your strategy for locating these blogs.

Compare and contrast the results from the 98 "random" blogs and the 98 "targeted" blogs.

Answer

NOT ATTEMPTED

References

- [1] Atkins, Grant. "CS532 Assignment 8 Repository" Github. N.p., 23 March 2017. Web. 23 March 2017. <https://github.com/grantat/cs532-s17/tree/master/assignments/A8>.
- [2] Richardson, Leonard. "Beautiful Soup Documentation." Beautiful Soup Documentation - Beautiful Soup 4.4.0 Documentation. N.p., n.d. Web. 24 Jan. 2017. <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>.
- [3] Segaran, Toby. "Programming Collective Intelligence". O' Reilly, 2007. Web. 6 April 2017. <https://github.com/arthur-e/Programming-Collective-Intelligence>.