2017

Assignment 8

CS 432

MIRANDA SMITH

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*

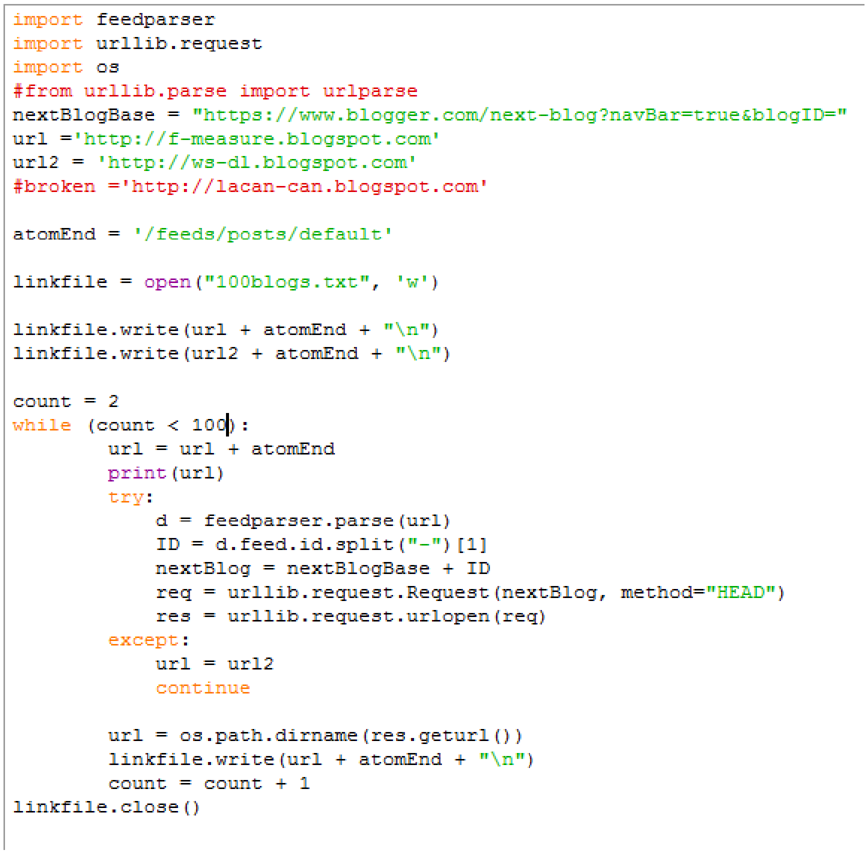
to github your code for grabbing the blogs and provide a list of blog URIs, both in the report and in github. Use the blog title as the identifier for each blog (and row of the

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.*

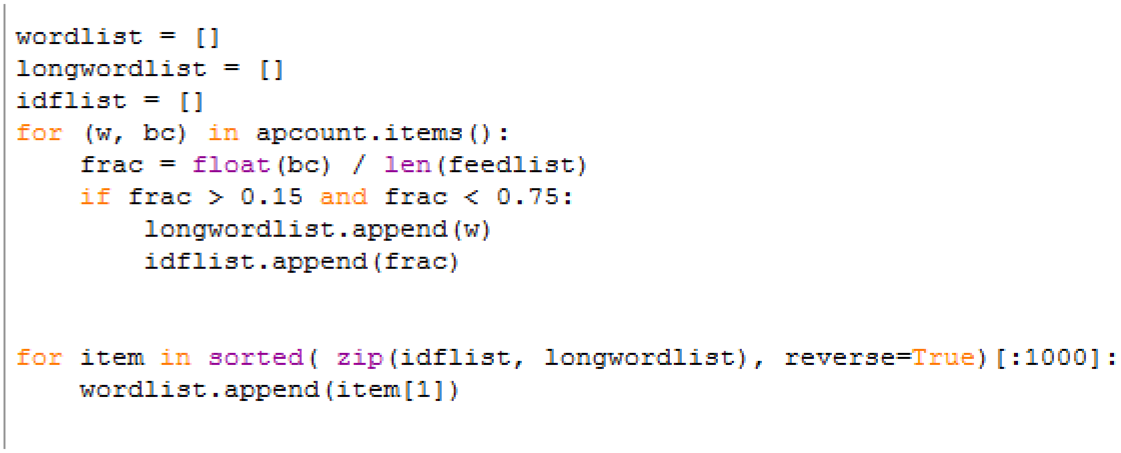
I created a program ‘nextBlog.py’ to collect 100 blogs from blogspot.com, including the 2 required ones. It uses the URL base for the Next Blog button and attaches the current blog ID. If something is wrong, either the next blog doesn’t exist and gives you a Go Daddy page, or the url won’t open at all, it restarts from one of the beginning URLs. That works because that links gives a random blog, there is no order. All Blogs are listed below and in ‘100blogs.txt’. There are slightly more blogs than 100 because some are in another language and will not be parsed by the next program and is unable to be located by the blog title. However, that server does not return a language header, so the most convenient method was to edit it to append to the file and add a few more.

[Intentionally left blank]



```
import feedparser
import urllib.request
import os
#from urllib.parse import urlparse
nextBlogBase = "https://www.blogger.com/next-blog?navBar=true&blogID="
url ='http://f-measure.blogspot.com'
url2 = 'http://ws-dl.blogspot.com'
#broken ='http://lacan-can.blogspot.com'

atomEnd = '/feeds/posts/default'

linkfile = open("100blogs.txt", 'w')

linkfile.write(url + atomEnd + "\n")
linkfile.write(url2 + atomEnd + "\n")

count = 2
while (count < 100):
        url = url + atomEnd
        print(url)
        try:
            d = feedparser.parse(url)
            ID = d.feed.id.split("-")[1]
            nextBlog = nextBlogBase + ID
            req = urllib.request.Request(nextBlog, method="HEAD")
            res = urllib.request.urlopen(req)
        except:
            url = url2
            continue

        url = os.path.dirname(res.geturl())
        linkfile.write(url + atomEnd + "\n")
        count = count + 1
linkfile.close()
```

I used the program 'generatefeedvector.py' from the Programming Collective Intelligence book to create the matrix. I will not put the whole code into the report because of its length, however it will be in the same directory on GitHub. I included the title and the summary to increase the amount of words to choose from because with only the title it wasn't anywhere close to 1000 words. The question wanted me to use TFIDF to choose the terms however, I found it prudent to not calculate the TF because that is a different score for each page and it didn't make sense to use that to pick words across the whole



```
wordlist = []
longwordlist = []
idflist = []
for (w, bc) in apcount.items():
    frac = float(bc) / len(feedlist)
    if frac > 0.15 and frac < 0.75:
        longwordlist.append(w)
        idflist.append(frac)


for item in sorted( zip(idflist, longwordlist), reverse=True)[:1000]:
    wordlist.append(item[1])
```

collection. I stuck with what they did in the book, with some small modifications to pick the best 1000.

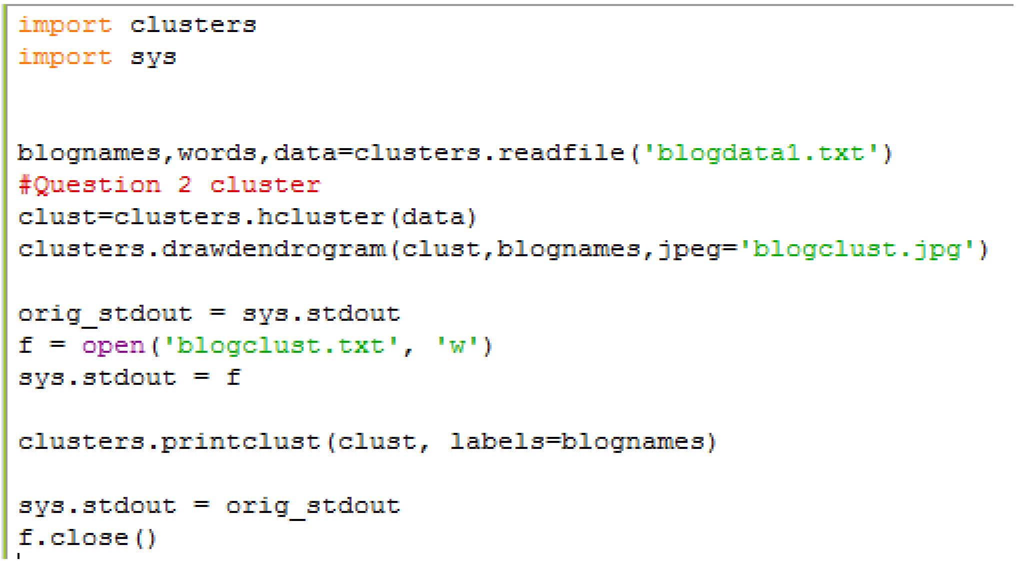
I choose those fractions based on a quick histogram I did in excel of all the fractions. The max was 0.87 and the lowest was 0. After running the program a few times to see the words chosen with the top limit, I decided on 0.75 because it removed most of the stop words without cutting into important words. From there I found that 0.15 was the largest number I could use that would include just over 1000 words.

The matrix is found in ‘blogdata1.txt’.

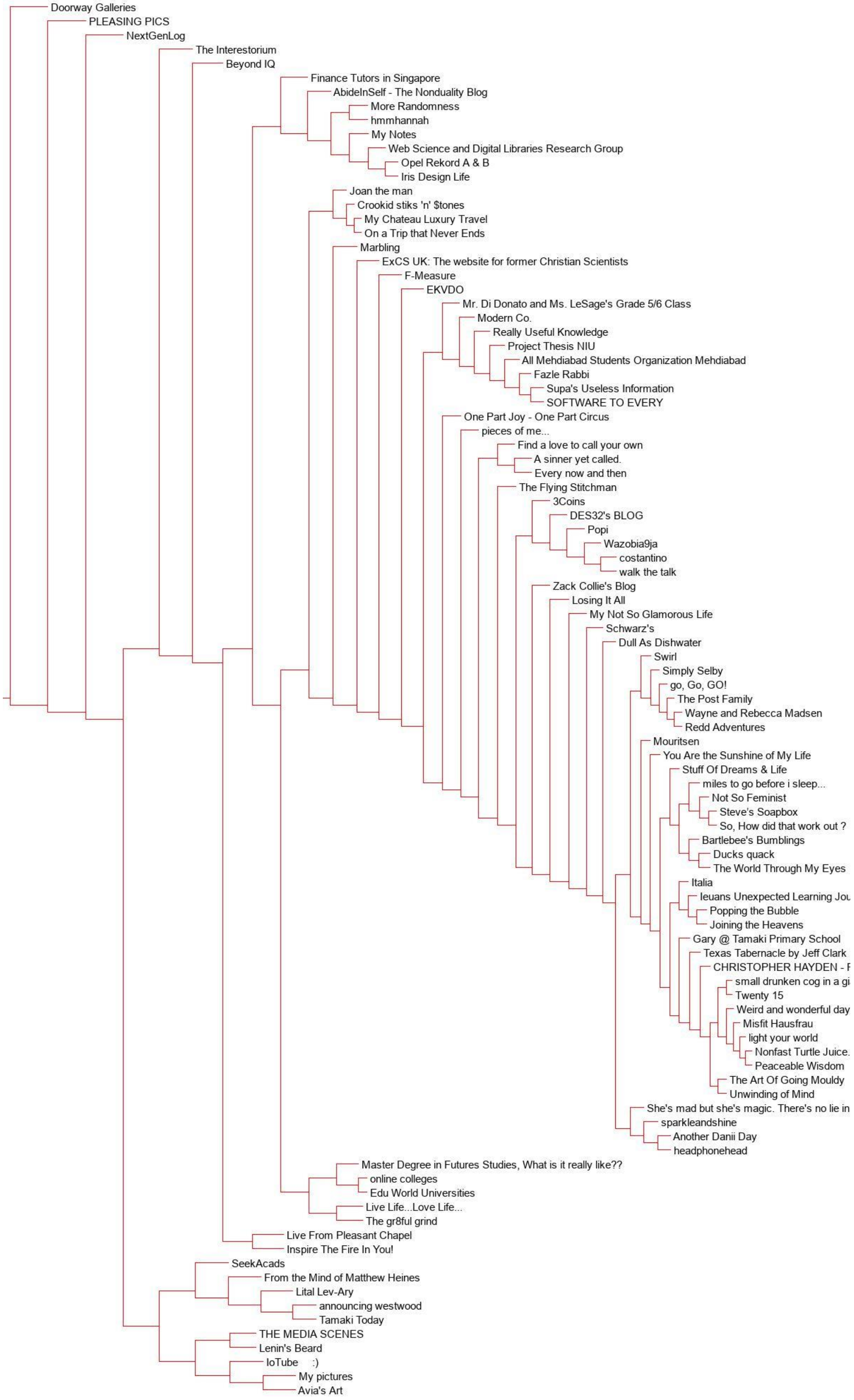
Question 2

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).

I used the program ‘createdendrogram.py’ to call the functions from ‘clusters.py’ which is a program found in the Programming Collective Intelligence book. Clusters had small modifications to make it compatible with python 3 but nothing to change the functionality. I set the standard output to a file because doing so in the function would be problematic since it is recursive. The created dendrogram is found in ‘blogclust.jpg’ and ‘blogclust.txt’. The picture is included on the next page.



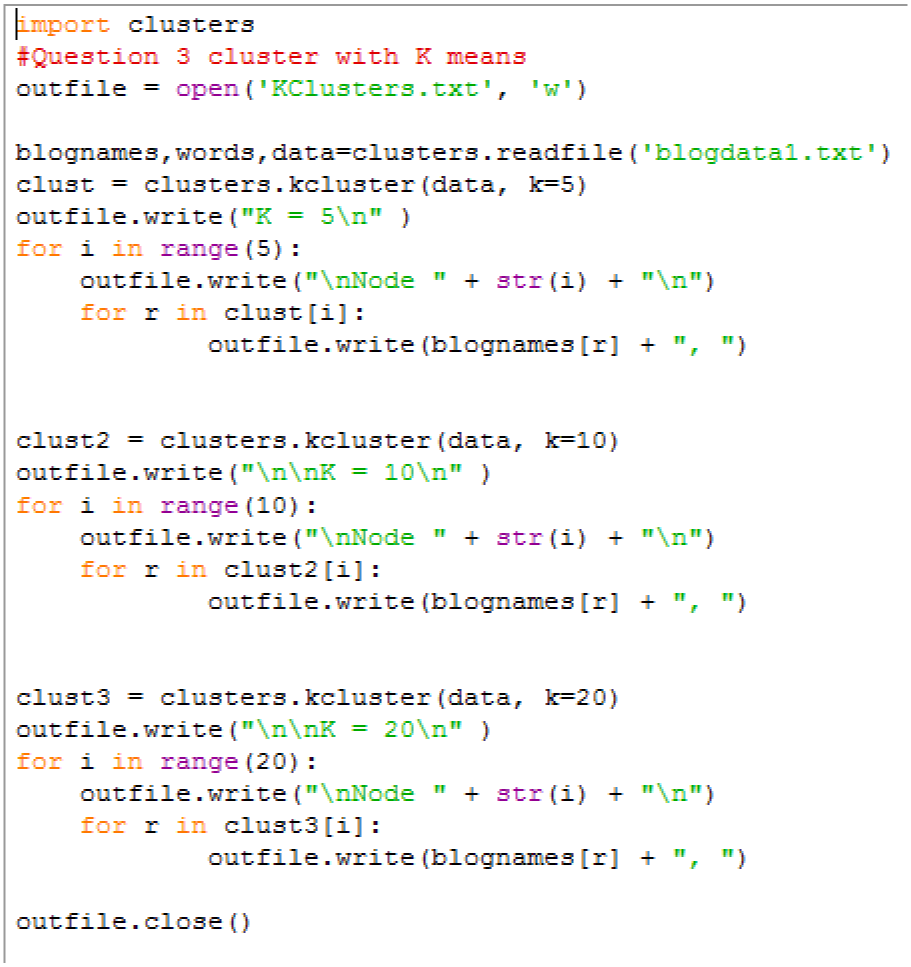
```
import clusters
import sys


blognames,words,data=clusters.readfile('blogdata1.txt')
#Question 2 cluster
clust=clusters.hcluster(data)
clusters.drawdendrogram(clust,blognames,jpeg='blogclust.jpg')

orig_stdout = sys.stdout
f = open('blogclust.txt', 'w')
sys.stdout = f

clusters.printclust(clust, labels=blognames)

sys.stdout = orig_stdout
f.close()
```


Doorway Galleries
PLEASING PICS
NextGenLog
The Interestorium
Beyond IQ
Finance Tutors in Singapore
AbideInSelf - The Nonduality Blog
More Randomness
hmmhannah
My Notes
Web Science and Digital Libraries Research Group
Opel Rekord A & B
Iris Design Life
Joan the man
Crookid stiks 'n' $tones
My Chateau Luxury Travel
On a Trip that Never Ends
Marbling
ExCS UK: The website for former Christian Scientists
F-Measure
EKVDO
Mr. Di Donato and Ms. LeSage's Grade 5/6 Class
Modern Co.
Really Useful Knowledge
Project Thesis NIU
All Mehdiabad Students Organization Mehdiabad
Fazle Rabbi
Supa's Useless Information
SOFTWARE TO EVERY
One Part Joy - One Part Circus
pieces of me...
Find a love to call your own
A sinner yet called.
Every now and then
The Flying Stitchman
3Coins
DES32's BLOG
Popi
Wazobia9ja
costantino
walk the talk
Zack Collie's Blog
Losing It All
My Not So Glamorous Life
Schwarz's
Dull As Dishwater
Swirl
Simply Selby
go, Go, GO!
The Post Family
Wayne and Rebecca Madsen
Redd Adventures
Mouritsen
You Are the Sunshine of My Life
Stuff Of Dreams & Life
miles to go before i sleep...
Not So Feminist
Steve's Soapbox
So, How did that work out ?
Bartlebee's Bumblings
Ducks quack
The World Through My Eyes
Italia
Ieuans Unexpected Learning Jou
Popping the Bubble
Joining the Heavens
Gary @ Tamaki Primary School
Texas Tabernacle by Jeff Clark
CHRISTOPHER HAYDEN - F
small drunken cog in a gi
Twenty 15
Weird and wonderful day
Misfit Hausfrau
light your world
Nonfast Turtle Juice.
Peaceable Wisdom
The Art Of Going Mouldy
Unwinding of Mind
She's mad but she's magic. There's no lie in
sparkleandshine
Another Danii Day
headphonehead
Master Degree in Futures Studies, What is it really like??
online colleges
Edu World Universities
Live Life...Love Life...
The gr8ful grind
Live From Pleasant Chapel
Inspire The Fire In You!
SeekAcads
From the Mind of Matthew Heines
Lital Lev-Ary
announcing westwood
Tamaki Today
THE MEDIA SCENES
Lenin's Beard
IoTube :)
My pictures
Avia's Art

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?

I created the program ‘createKcluster.py’ to call the function kcluster() in the ‘cluster.py’ program and properly present the output. Since the output is long is will not be included in this report but it is in ‘KClusters.txt’



```
import clusters
#Question 3 cluster with K means
outfile = open('KClusters.txt', 'w')

blognames,words,data=clusters.readfile('blogdata1.txt')
clust = clusters.kcluster(data, k=5)
outfile.write("K = 5\n" )
for i in range(5):
    outfile.write("\nNode " + str(i) + "\n")
    for r in clust[i]:
            outfile.write(blognames[r] + ", ")


clust2 = clusters.kcluster(data, k=10)
outfile.write("\n\nK = 10\n" )
for i in range(10):
    outfile.write("\nNode " + str(i) + "\n")
    for r in clust2[i]:
            outfile.write(blognames[r] + ", ")


clust3 = clusters.kcluster(data, k=20)
outfile.write("\n\nK = 20\n" )
for i in range(20):
    outfile.write("\nNode " + str(i) + "\n")
    for r in clust3[i]:
            outfile.write(blognames[r] + ", ")

outfile.close()
```

Iteration count for each k value per the program system output:

K=5 took 11 iterations

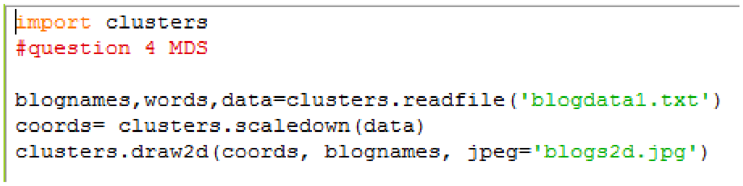
K= 10 took 9 iteration

K=20 took 6 iterations

Question 4

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

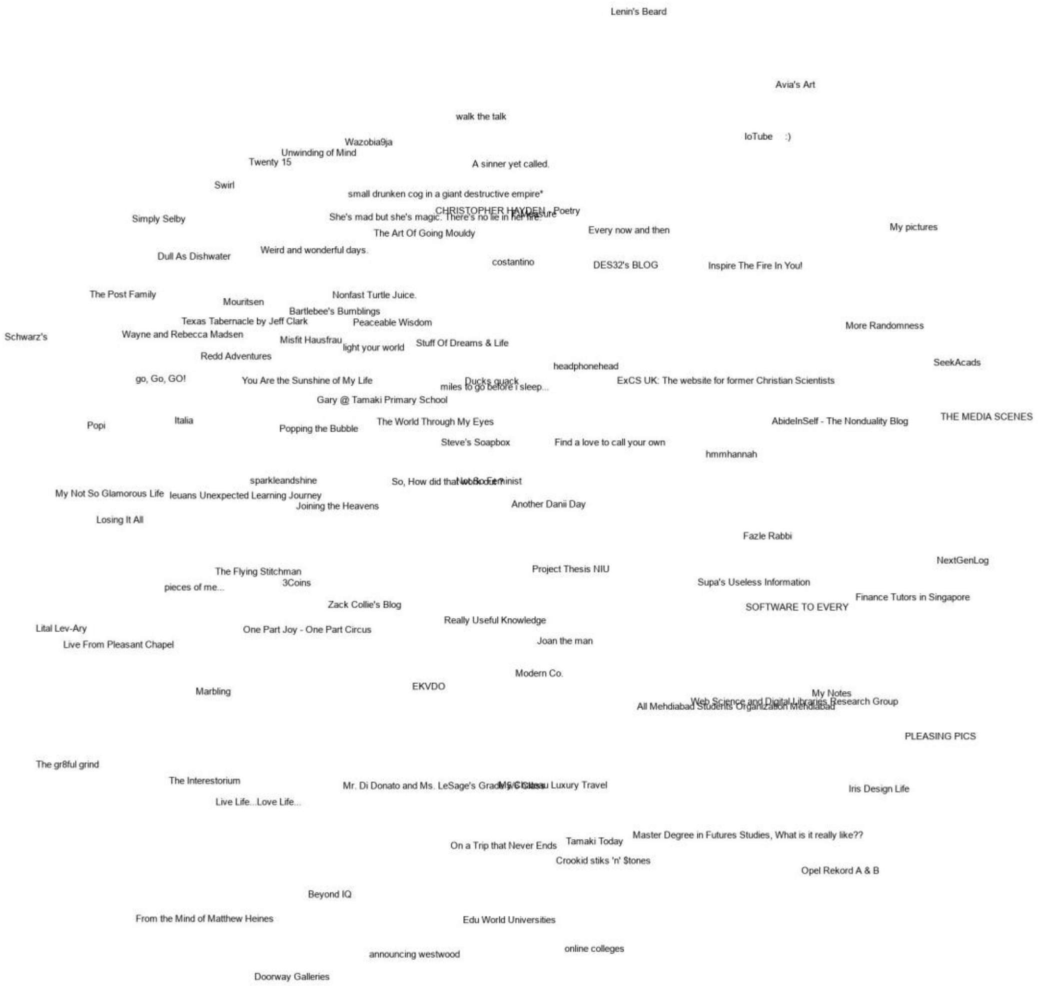
I created the program ‘MDS.py’ to call the functions scaledown() and draw2d() in the ‘cluster.py’ program.



```
import clusters
#question 4 MDS

blognames,words,data=clusters.readfile('blogdata1.txt')
coords= clusters.scaledown(data)
clusters.draw2d(coords, blognames, jpeg='blogs2d.jpg')
```

The error went from a max of 4357.900447473182 to a minimum of 2501.438879265835 in 190 iterations, including the 1 it took to backtrack when the error started increasing.

The output is in ‘blogs2d.jpg’ which is included on the following page.


Lenin's Beard
Avia's Art
walk the talk
IoTube :)
Wazobia9ja
Unwinding of Mind
Twenty 15
A sinner yet called.
Swirl
small drunken cog in a giant destructive empire*
CHRISTOPHER HAYDEN - Poetry
She's mad but she's magic. There's no lie in her fire.
Simply Selby
My pictures
The Art Of Going Mouldy
Every now and then
Weird and wonderful days.
Dull As Dishwater
costantino
DES32's BLOG
Inspire The Fire In You!
The Post Family
Nonfast Turtle Juice.
Mouritsen
Bartlebee's Bumblings
Texas Tabernacle by Jeff Clark
Peaceable Wisdom
More Randomness
Schwarz's
Wayne and Rebecca Madsen
Misfit Hausfrau
light your world
Stuff Of Dreams & Life
Redd Adventures
headphonehead
SeekAcads
go, Go, GO!
You Are the Sunshine of My Life
Ducks quack
miles to go before i sleep...
ExCS UK: The website for former Christian Scientists
Gary @ Tamaki Primary School
Popi
Italia
Popping the Bubble
The World Through My Eyes
AbideInSelf - The Nonduality Blog
THE MEDIA SCENES
Steve's Soapbox
Find a love to call your own
hmmhannah
sparkleandshine
So, How did that work out?
Not So Feminist
My Not So Glamorous Life
Ieuans Unexpected Learning Journey
Joining the Heavens
Another Danii Day
Losing It All
Fazle Rabbi
NextGenLog
The Flying Stitchman
Project Thesis NIU
pieces of me...
3Coins
Supa's Useless Information
Finance Tutors in Singapore
Zack Collie's Blog
SOFTWARE TO EVERY
Really Useful Knowledge
Lital Lev-Ary
One Part Joy - One Part Circus
Live From Pleasant Chapel
Joan the man
Modern Co.
EKVDO
Marbling
My Notes
Web Science and Digital Libraries Research Group
All Mehdiabad Students Organization Mehdiabad
PLEASING PICS
The gr8ful grind
The Interestorium
Mr. Di Donato and Ms. LeSage's Grade 5/6 Class
Luxury Travel
Iris Design Life
Live Life...Love Life...
Master Degree in Futures Studies, What is it really like??
Tamaki Today
On a Trip that Never Ends
Crookid stiks 'n' $tones
Opel Rekord A & B
Beyond IQ
From the Mind of Matthew Heines
Edu World Universities
online colleges
announcing westwood
Doorway Galleries