**Assignment 8**
CS532, Web Science, Spring 2017
Old Dominion University, Computer Science Dept

**Hussam Hallak**
CS Master's Student
Prof: Dr. Nelson

## 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:

The approach is divided into three steps:

1. The first step is getting the urls for the blogs. I wrote a small python script, "getblogurls.py" to save both blog links provided in the assignment question, and to collect 98 random blogs. I collected more than 98 because I wanted to make sure that I have 100 unique blogs at the end. I ran the output file "urls.txt" through a python script I wrote for a previous assignment to remove duplicated urls. I finally used "head" command to grab the exact amount of blogs I need, 100 blogs, from the top of the file. I saved the final result in a file named "100blogs.txt".

Listing 1: The content of getblogurls.py

```
import sys
from bs4 import BeautifulSoup
import urllib2
import re

fh_output = open('urls.txt','w')
fh_output.write('http://f-measure.blogspot.com/'+'\n')
fh_output.write('http://ws-dl.blogspot.com/'+'\n')

for i in range(200):
  try:
    url = 'http://www.blogger.com/next-blog?navBar=true&blogID
       =3471633091411211117'
    html_page = urllib2.urlopen(url)
    html = html_page.read()
    soup = BeautifulSoup(html, "html.parser")
    for link in soup.find_all('link'):
      if link['rel']==['alternate'] and link['type']=='application/atom+xml':
            blog_url = link['href']
            blog_url = blog_url[:-19]
            fh_output.write(blog_url+'\n')
   except:
      continue
fh_output.close()
```

After running the script and the command mentioned earlier, I got the following output:

Listing 2: Running getblogurls.py

```
root@ima-app:/var/www/Hussam/A8# python getblogurls.py
root@ima-app:/var/www/Hussam/A8# cat urls.txt | wc -l
189
root@ima-app:/var/www/Hussam/A8# python makeunique.py urls.txt uniqueurls.txt
root@ima-app:/var/www/Hussam/A8# cat uniqueurls.txt | wc -l
115
root@ima-app:/var/www/Hussam/A8# head -n -15 uniqueurls.txt > 100blogs.txt
root@ima-app:/var/www/Hussam/A8# cat 100blogs.txt |wc -l
100
root@ima-app:/var/www/Hussam/A8# cat 100blogs.txt
http://f-measure.blogspot.com/
http://ws-dl.blogspot.com/
http://my-name-is-blue-canary.blogspot.com/
http://fridaynightdream.blogspot.com/
http://nathaliealves.blogspot.com/
http://stephanieveto.blogspot.com/
http://dcresider.blogspot.com/
http://mobbie2.blogspot.com/
http://nonsensealamode.blogspot.com/
http://fractalpress.blogspot.com/
http://ablazingflame.blogspot.com/
http://pithytitlehere.blogspot.com/
http://angie-dynamo.blogspot.com/
http://revolverusa.blogspot.com/
http://steel-city-rust.blogspot.com/
http://www.hipindetroit.com/
http://markeortega.blogspot.com/
http://ilovetotaldestruction.blogspot.com/
http://sixtyat60.blogspot.com/
http://beyondthepond-wpl.blogspot.com/
http://doyouneedatv.blogspot.com/
http://franbrighton.blogspot.com/
http://machineryofdenial.blogspot.com/
https://urockradio.blogspot.com/
http://guardtheguardians.blogspot.com/
http://davecromwellwrites.blogspot.com/
http://maggotcaviar.blogspot.com/
http://storiesfromthecityradiovalencia.blogspot.com/
http://thehubkxci.blogspot.com/
http://www.sonology.com/
http://adrianomarquesblog.blogspot.com/
http://truthfulmood.blogspot.com/
http://chantellesmedia2.blogspot.com/
http://lost-places-hamburg.blogspot.com/
```

```
http://cherryarea.blogspot.com/
http://www.thestarkonline.com/
https://norecordshopsleft.blogspot.com/
http://blog.spinitron.com/
http://mediastudiesa2advanced.blogspot.com/
http://ps-music.blogspot.com/
http://bonjourgirl.blogspot.com/
http://dpl2blog.blogspot.com/
http://ohyesjonsi.blogspot.com/
http://musicneedshelp.blogspot.com/
http://lostintheshuffle899.blogspot.com/
http://hiiijaaackie.blogspot.com/
http://www.holaolamusic.com/
http://itll-glow-on-you.blogspot.com/
http://semregrasluispink.blogspot.com/
http://theidealcopy.blogspot.com/
http://onestunningsingleegg.blogspot.com/
http://mts-dailythemes.blogspot.com/
http://bogglemethursday.blogspot.com/
http://macthemost.blogspot.com/
http://mesastivromia.blogspot.com/
http://floorshimezipperboots.blogspot.com/
http://jasminehodge.blogspot.com/
http://theonionfield.blogspot.com/
http://bleakbliss.blogspot.com/
http://flipmpip.blogspot.com/
http://duchessnonetheless.blogspot.com/
http://dinosaursarefun.blogspot.com/
http://stonehillsketchbook.blogspot.com/
http://skinnyshoes.blogspot.com/
http://spicyseatdolphin.blogspot.com/
http://travelingneighborhood.blogspot.com/
http://somecallitnoise.blogspot.com/
http://elijace.blogspot.com/
http://cuzmusicrocks.blogspot.com/
http://organmyth.blogspot.com/
http://thetremagazine.blogspot.com/
http://johnandmaureensanto.blogspot.com/
http://campusbuzzwsou.blogspot.com/
http://hani-bittersweet.blogspot.com/
http://rantsfromthepants.blogspot.com/
http://simonegoes.blogspot.com/
http://ourstatus.blogspot.com/
http://momentarilymusical.blogspot.com/
http://www.gypsyrhapsody.com/
http://jlmdlhlcm1516.blogspot.com/
http://didnotchart.blogspot.com/
http://naoponhomusica.blogspot.com/
http://castironsongs.blogspot.com/
http://psychfolkmusic.blogspot.com/
http://superchicken46.blogspot.com/
http://paulinag-mediaa2.blogspot.com/
http://sixeyes.blogspot.com/
http://justplayingfavorites.blogspot.com/
http://markfishers-musicreview.blogspot.com/
http://encorenorthernireland.blogspot.com/
```

```
http://mtjrrantsravesonmusic.blogspot.com/
http://myopiamuse.blogspot.com/
http://alayerofchips.blogspot.com/
http://barakoffein.blogspot.com/
http://makeupmusicandfashion.blogspot.com/
http://glipress.blogspot.com/
http://mileinmine.blogspot.com/
http://kidchair.blogspot.com/
http://mandolinnn.blogspot.com/
http://themusicbinge.blogspot.com/
root@ima-app:/var/www/Hussam/A8#
```

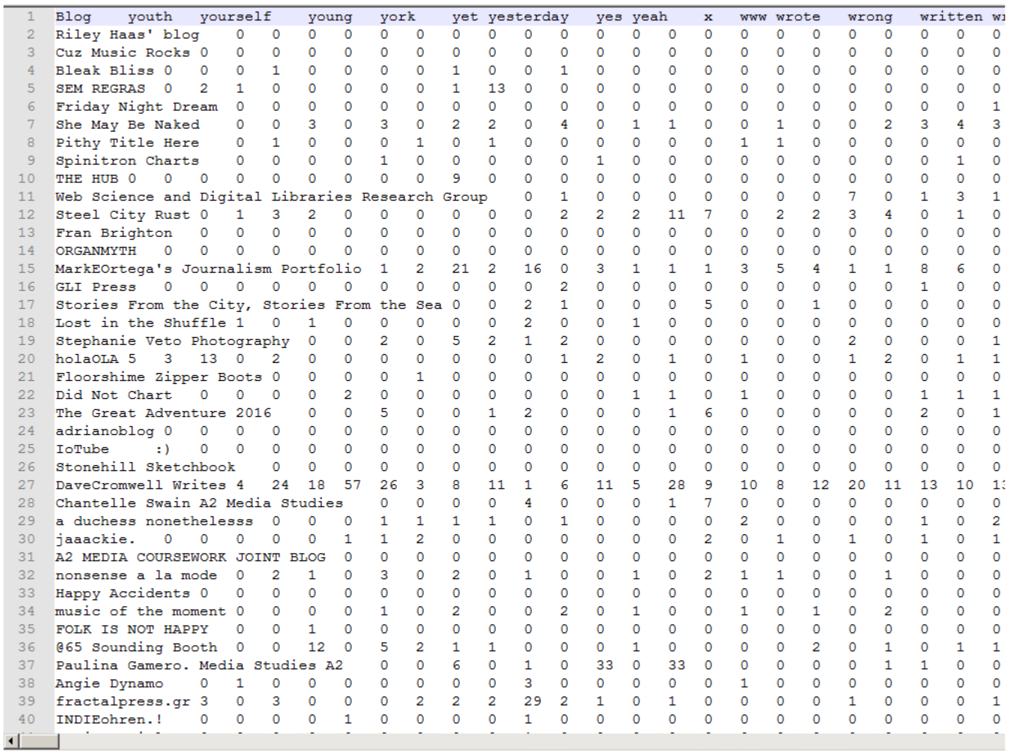
2. The next step is getting all pages for each blog we collected. I wrote a python script "getpages.py" to do that. It takes the file "100blogs.txt" as input, command line argument. The script grabs all pages for each blog in the file. The output is saved in a file named "pages.txt". The content of the file "pages.txt" is too big to include in the report, but it is included in the "Q1" folder.

Listing 3: The content of getpages.py

```python
import sys
from bs4 import BeautifulSoup
import urllib2
import re
if len(sys.argv) < 2:
   print "Usage: python getpages.py <input_file>"
   print "e.g: python getpages.py 100blogs.txt"
   exit()
fh_output = open('pages.txt', 'w')

def getNextPage(link):
   try:
      html = urllib2.urlopen(link).read()
      soup = BeautifulSoup(html, 'lxml')
      next_page = soup.find('link', rel="next")
      if(next_page != []):
         next_page = next_page.get('href')
         return next_page
   except:
      return False

def getAllPages(link):
   all_pages = []
   next_page = getNextPage(link)
   while(next_page != False):
      all_pages.append(next_page)
      next_page = getNextPage(next_page)
   return all_pages

for blog in open(sys.argv[1], 'r'):
   pages = []
   try:
         html = urllib2.urlopen(blog).read()
         soup = BeautifulSoup(html, 'lxml')
      title = soup.title.string.encode('ascii')
      rss = soup.find('link', type='application/atom+xml')
```

```
        rss = rss.get('href')
        pages = getAllPages(rss)
        pages.insert(0,rss)
        for page in pages:
            fh_output.write(page + '\n')
    except:
        continue
fh_output.close()
```

3. The last step is to create a blog-term matrix from the blogs' pages we collected in step 2. I used the script provided by "PCI" textbook. The file "pages.txt" is taken as input from the command line. The output, the desired blog-term matrix, is saved to the file "blogdata.txt". The content of the file "blogdata.txt" is too big to include in the report, but it is included in the "Q1" folder. Here is a screen shot of the file "blogdata.txt":



```
Blog    youth   yourself    young   york    yet yesterday   yes yeah    x   www wrote   wrong   written w
Riley Haas' blog    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
Cuz Music Rocks 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
Bleak Bliss 0   0   0   1   0   0   0   0   1   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0
SEM REGRAS  0   2   1   0   0   0   0   0   1   13  0   0   0   0   0   0   0   0   0   0   0   0   0   0
Friday Night Dream  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   1
She May Be Naked    0   0   3   0   3   0   2   2   0   4   0   1   1   0   0   1   0   0   2   3   4   3
Pithy Title Here    0   1   0   0   0   1   0   1   0   0   0   0   0   0   1   1   0   0   0   0   0   0
Spinitron Charts    0   0   0   0   1   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   1   0
THE HUB 0   0   0   0   0   0   0   0   0   9   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
Web Science and Digital Libraries Research Group    0   1   0   0   0   0   0   0   0   7   0   1   3   1
Steel City Rust 0   1   3   2   0   0   0   0   0   0   2   2   2   11  7   0   2   2   3   4   0   1   0
Fran Brighton   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
ORGANMYTH   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
MarkEOrtega's Journalism Portfolio  1   2   21  2   16  0   3   1   1   1   3   5   4   1   1   8   6   0
GLI Press   0   0   0   0   0   0   0   0   0   0   0   2   0   0   0   0   0   0   0   0   0   1   0   0
Stories From the City, Stories From the Sea 0   0   2   1   0   0   0   5   0   0   1   0   0   0   0   0
Lost in the Shuffle 1   0   1   0   0   0   0   0   2   0   0   1   0   0   0   0   0   0   0   0   0   0
Stephanie Veto Photography  0   0   2   0   5   2   1   2   0   0   0   0   0   0   0   2   0   0   0   1
holaOLA 5   3   13  0   2   0   0   0   0   0   0   0   1   2   0   1   0   1   0   0   1   2   0   1   1
Floorshime Zipper Boots 0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
Did Not Chart   0   0   0   0   2   0   0   0   0   0   0   0   1   1   0   1   0   0   0   0   1   1   1
The Great Adventure 2016    0   0   5   0   0   1   2   0   0   0   1   6   0   0   0   0   0   2   0   1
adrianoblog 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
IoTube     :)   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
Stonehill Sketchbook    0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
DaveCromwell Writes 4   24  18  57  26  3   8   11  1   6   11  5   28  9   10  8   12  20  11  13  10  1
Chantelle Swain A2 Media Studies    0   0   0   0   4   0   0   0   1   7   0   0   0   0   0   0   0   0
a duchess nonethelesss  0   0   0   1   1   1   1   0   1   0   0   0   0   2   0   0   0   0   1   0   2
jaaackie.   0   0   0   0   0   1   1   2   0   0   0   0   0   0   0   2   0   1   0   1   0   1   0   1
A2 MEDIA COURSEWORK JOINT BLOG  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
nonsense a la mode  0   2   1   0   3   0   2   0   1   0   0   1   0   2   1   1   0   0   1   0   0   0
Happy Accidents 0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
music of the moment 0   0   0   0   1   0   2   0   0   2   0   1   0   0   1   0   1   0   2   0   0   0
FOLK IS NOT HAPPY   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
@65 Sounding Booth  0   0   12  0   5   2   1   1   0   0   0   1   0   0   0   0   2   0   1   0   1   1
Paulina Gamero. Media Studies A2    0   0   6   0   1   0   33  0   33  0   0   0   0   0   1   1   0   0
Angie Dynamo    0   1   0   0   0   0   0   0   0   3   0   0   0   0   0   1   0   0   0   0   0   0   0
fractalpress.gr 3   0   3   0   0   0   2   2   2   29  2   1   0   1   0   0   0   0   1   0   0   0   1
INDIEohren.!    0   0   0   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0
```

## Included Files:

100blogs.txt, blogdata.txt, generatefeedvector.py, getblogurls.py, getpages.py, makeunique.py, pages.txt, uniqueurls.txt, urls.txt, blogdata.png

## 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:

In order to create an ASCII and JPEG dendogram, I used the script provided by “PCI” textbook and saved it to a file “makedend.py” to run it.

Listing 4: The content of makedend.py

```
from math import sqrt
from PIL import Image,ImageDraw

def readfile(filename):
    lines=[line for line in file(filename)]
    colnames=lines[0].strip( ).split('\t')[1:]
    rownames=[]
    data=[]
    for line in lines[1:]:
        p=line.strip( ).split('\t')
        # First column in each row is the rowname
        rownames.append(p[0])
        # The data for this row is the remainder of the row
        data.append([float(x) for x in p[1:]])
    return rownames,colnames,data


def pearson(v1,v2):
    # Simple sums
    sum1=sum(v1)
    sum2=sum(v2)

    # Sums of the squares
    sum1Sq=sum([pow(v,2) for v in v1])
    sum2Sq=sum([pow(v,2) for v in v2])

    # Sum of the products
    pSum=sum([v1[i]*v2[i] for i in range(len(v1))])

    # Calculate r (Pearson score)
    num=pSum-(sum1*sum2/len(v1))
    den=sqrt((sum1Sq-pow(sum1,2)/len(v1))*(sum2Sq-pow(sum2,2)/len(v1)))
    if den==0: return 0

    return 1.0-num/den


class bicluster:
    def __init__(self,vec,left=None,right=None,distance=0.0,id=None):
        self.left=left
        self.right=right
        self.vec=vec
        self.id=id
        self.distance=distance


def hcluster(rows,distance=pearson):
    distances={}
    currentclustid=-1
    # Clusters are initially just the rows
    clust=[bicluster(rows[i],id=i) for i in range(len(rows))]

    while len(clust)>1:
        lowestpair=(0,1)
        closest=distance(clust[0].vec,clust[1].vec)

        # loop through every pair looking for the smallest distance
        for i in range(len(clust)):
            for j in range(i+1,len(clust)):
                # distances is the cache of distance calculations
                if (clust[i].id,clust[j].id) not in distances:
                    distances[(clust[i].id,clust[j].id)]=distance(clust[i].vec,
                        clust[j].vec)

                d=distances[(clust[i].id,clust[j].id)]

                if d<closest:
                    closest=d
                    lowestpair=(i,j)

        # calculate the average of the two clusters
        mergevec=[
        (clust[lowestpair[0]].vec[i]+clust[lowestpair[1]].vec[i])/2.0
        for i in range(len(clust[0].vec))]

        # create the new cluster
        newcluster=bicluster(mergevec,left=clust[lowestpair[0]],
                             right=clust[lowestpair[1]],
                             distance=closest,id=currentclustid)

        # cluster ids that weren't in the original set are negative
        currentclustid-=1
        del clust[lowestpair[1]]
        del clust[lowestpair[0]]
        clust.append(newcluster)

    return clust[0]


def printclust(clust,labels=None,n=0):
    # indent to make a hierarchy layout
    for i in range(n): print ' ',
    if clust.id<0:
        # negative id means that this is branch
        print ('-')
    else:
        # positive id means that this is an endpoint
        if labels==None: print (clust.id)
        else: print (labels[clust.id])

    # now print the right and left branches
    if clust.left!=None: printclust(clust.left,labels=labels,n=n+1)
    if clust.right!=None: printclust(clust.right,labels=labels,n=n+1)
```

```python
def getheight(clust):
    # Is this an endpoint? Then the height is just 1
    if clust.left==None and clust.right==None: return 1

    # Otherwise the height is the same of the heights of
    # each branch
    return getheight(clust.left)+getheight(clust.right)


def getdepth(clust):
    # The distance of an endpoint is 0.0
    if clust.left==None and clust.right==None: return 0

    # The distance of a branch is the greater of its two sides
    # plus its own distance
    return max(getdepth(clust.left),getdepth(clust.right))+clust.distance


def drawdendrogram(clust,labels,jpeg='clusters.jpg'):
    # height and width
    h=getheight(clust)*20
    w=1200
    depth=getdepth(clust)

    # width is fixed, so scale distances accordingly
    scaling=float(w-150)/depth

    # Create a new image with a white background
    img=Image.new('RGB',(w,h),(255,255,255))
    draw=ImageDraw.Draw(img)

    draw.line((0,h/2,10,h/2),fill=(255,0,0))

    # Draw the first node
    drawnode(draw,clust,10,(h/2),scaling,labels)
    img.save(jpeg,'JPEG')


def drawnode(draw,clust,x,y,scaling,labels):
    if clust.id<0:
        h1=getheight(clust.left)*20
        h2=getheight(clust.right)*20
        top=y-(h1+h2)/2
        bottom=y+(h1+h2)/2
        # Line length
        ll=clust.distance*scaling
        # Vertical line from this cluster to children
        draw.line((x,top+h1/2,x,bottom-h2/2),fill=(255,0,0))

        # Horizontal line to left item
        draw.line((x,top+h1/2,x+ll,top+h1/2),fill=(255,0,0))

        # Horizontal line to right item
        draw.line((x,bottom-h2/2,x+ll,bottom-h2/2),fill=(255,0,0))
        # Call the function to draw the left and right nodes
        drawnode(draw,clust.left,x+ll,top+h1/2,scaling,labels)
        drawnode(draw,clust.right,x+ll,bottom-h2/2,scaling,labels)
    else:
        # If this is an endpoint, draw the item label
        draw.text((x+5,y-7),labels[clust.id],(0,0,0))

blognames,words,data=readfile('blogdata.txt')
clust=hcluster(data)
printclust(clust,labels=blognames)
drawdendrogram(clust,blognames,jpeg='dend.jpg')
```

The script uses the file blogdata.txt to create the dendogram that clusters the most similar blogs. The ASCII dendogram is saved to the file “dend.txt” and the JPEG dendogram is saved to the file “dend.jpg”.

Listing 5: Running makedend.py

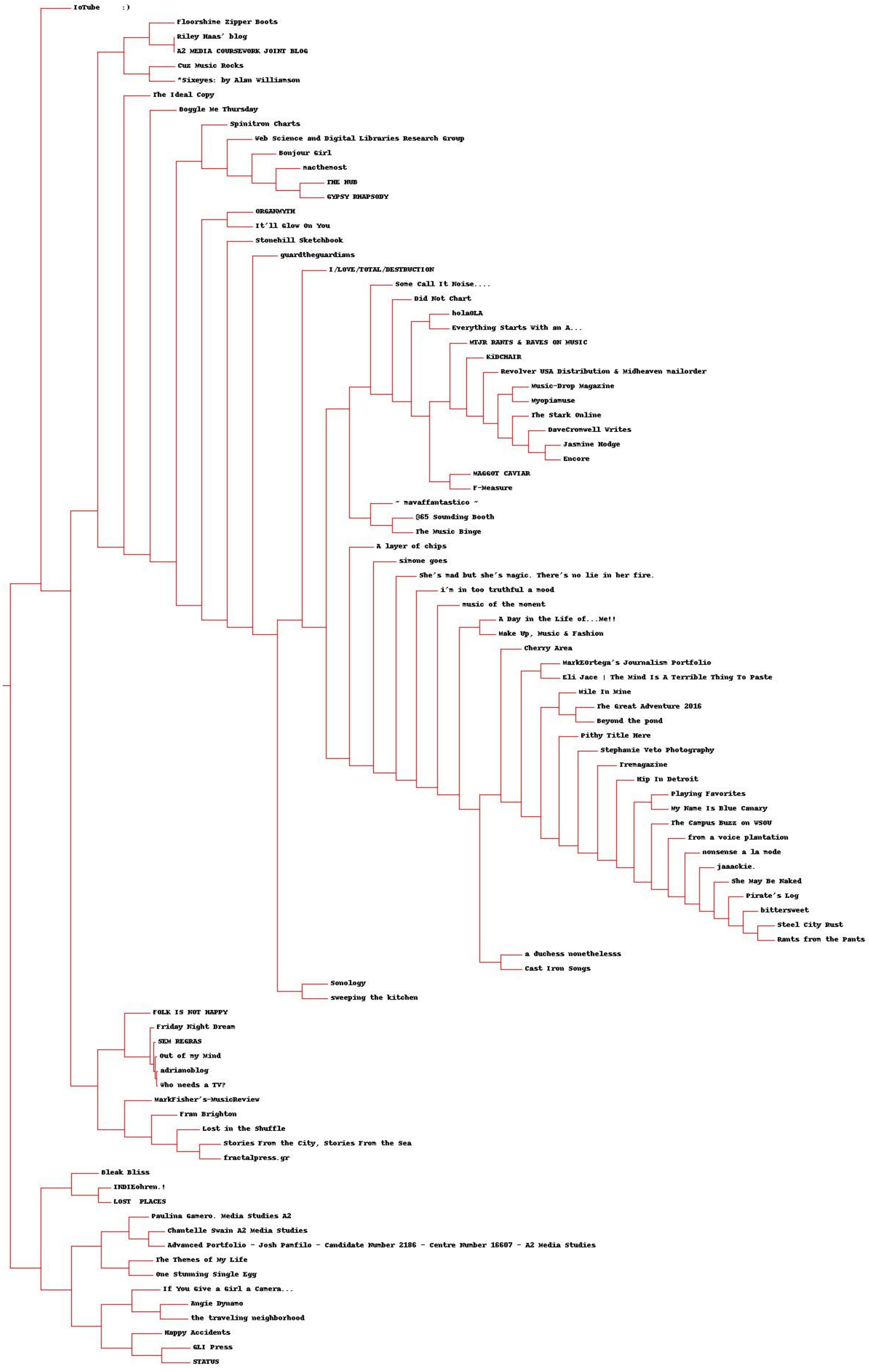
```
root@ima-app:/var/www/Hussam/A8# python makedend.py > dend.txt
root@ima-app:/var/www/Hussam/A8# ls
100blogs.txt dend.jpg generatefeedvector.py getpages.py makeunique.py uniqueurls.
    txt
blogdata.txt dend.txt getblogurls.py      makedend.py pages.txt    urls.txt
root@ima-app:/var/www/Hussam/A8# cat dend.txt
-
  -
    IoTube    :)
    -
      -
        -
          -
            Floorshime Zipper Boots
            -
              Riley Haas' blog
              A2 MEDIA COURSEWORK JOINT BLOG
          -
            Cuz Music Rocks
            *Sixeyes: by Alan Williamson
        -
          The Ideal Copy
          -
            Boggle Me Thursday
            -
              -
                Spinitron Charts
                -
                  Web Science and Digital Libraries Research Group
                  -
                    Bonjour Girl
                    -
                      macthemost
                      -
                        THE HUB
                        GYPSY RHAPSODY
              -
                -
```

```
 ORGANMYTH
 It’ll Glow On You
-
 Stonehill Sketchbook
 -
  guardtheguardians
  -
   -
    I/LOVE/TOTAL/DESTRUCTION
    -
     -
      -
       Some Call It Noise....
       -
        Did Not Chart
        -
         -
          holaOLA
          Everything Starts With an A...
         -
          -
           MTJR RANTS & RAVES ON MUSIC
           -
            KiDCHAIR
            -
             Revolver USA Distribution & Midheaven
               mailorder
             -
              -
               Music-Drop Magazine
               Myopiamuse
              -
               The Stark Online
               -
                DaveCromwell Writes
                -
                 Jasmine Hodge
                 Encore
          -
           MAGGOT CAVIAR
           F-Measure
      -
       ~ mavaffantastico ~
       -
        @65 Sounding Booth
        The Music Binge
     -
      A layer of chips
      -
       simone goes
       -
        She’s mad but she’s magic. There’s no lie in her
          fire.
        -
         i’m in too truthful a mood
         -
```

```
                              music of the moment
                              -
                                -
                                  A Day in the Life of...Me!!
                                  Make Up, Music & Fashion
                                -
                                  -
                                    Cherry Area
                                    -
                                      -
                                        MarkEOrtega’s Journalism Portfolio
                                        Eli Jace | The Mind Is A Terrible Thing
                                             To Paste
                                      -
                                        -
                                          Mile In Mine
                                          -
                                            The Great Adventure 2016
                                            Beyond the pond
                                        -
                                          Pithy Title Here
                                          -
                                            Stephanie Veto Photography
                                            -
                                              Tremagazine
                                              -
                                                Hip In Detroit
                                                -
                                                  -
                                                    Playing Favorites
                                                    My Name Is Blue Canary
                                                  -
                                                    The Campus Buzz on WSOU
                                                    -
                                                      from a voice plantation
                                                      -
                                                        nonsense a la mode
                                                        -
                                                          jaaackie.
                                                          -
                                                            She May Be Naked
                                                            -
                                                              Pirate’s Log
                                                              -
                                                                bittersweet
                                                                -
                                                                  Steel City Rust
                                                                  Rants from the
                                                                      Pants
                                  -
                                    a duchess nonethelesss
                                    Cast Iron Songs
                -
                  Sonology
                  sweeping the kitchen
-
```

```
        -
         FOLK IS NOT HAPPY
         -
          Friday Night Dream
          -
           SEM REGRAS
           -
            Out of my Mind
            -
             adrianoblog
             Who needs a TV?
        -
         MarkFisher's-MusicReview
         -
          Fran Brighton
          -
           Lost in the Shuffle
           -
            Stories From the City, Stories From the Sea
            fractalpress.gr
    -
     -
      Bleak Bliss
      -
       INDIEohren.!
       LOST PLACES
     -
      -
       -
        Paulina Gamero. Media Studies A2
        -
         Chantelle Swain A2 Media Studies
         Advanced Portfolio - Josh Pamfilo - Candidate Number 2186 - Centre
             Number 16607 - A2 Media Studies
       -
        The Themes of My Life
        One Stunning Single Egg
      -
       -
        If You Give a Girl a Camera...
        -
         Angie Dynamo
         the traveling neighborhood
       -
        Happy Accidents
        -
         GLI Press
         STATUS
root@ima-app:/var/www/Hussam/A8#
```

**JPEG dendrogram:**


IoTube :)
Floorshine Zipper Boots
Riley Maas' blog
A2 MEDIA COURSEWORK JOINT BLOG
Cuz Music Rocks
*Sixeyes: by Alan Williamson
The Ideal Copy
Boggle Me Thursday
Spinitron Charts
Web Science and Digital Libraries Research Group
Bonjour Girl
macthemost
THE HUB
GYPSY RHAPSODY
ORGANMYTH
It'll Glow On You
Stonehill Sketchbook
guardtheguardians
I/LOVE/TOTAL/DESTRUCTION
Some Call It Noise....
Did Not Chart
holaOLA
Everything Starts With an A...
MTJR RANTS & RAVES ON MUSIC
KiDCHAIR
Revolver USA Distribution & Midheaven mailorder
Music-Drop Magazine
Myopiamuse
The Stark Online
DaveCromwell Writes
Jasmine Hodge
Encore
MAGGOT CAVIAR
F-Measure
~ navaffantastico ~
@65 Sounding Booth
The Music Binge
A layer of chips
simone goes
She's mad but she's magic. There's no lie in her fire.
i'm in too truthful a mood
music of the moment
A Day in the Life of...Me!!
Make Up, Music & Fashion
Cherry Area
MarkEOrtega's Journalism Portfolio
Eli Jace | The Mind Is A Terrible Thing To Paste
Mile In Mine
The Great Adventure 2016
Beyond the pond
Pithy Title Here
Stephanie Veto Photography
Tremagazine
Hip In Detroit
Playing Favorites
My Name Is Blue Canary
The Campus Buzz on WSOU
from a voice plantation
nonsense a la mode
jaaackie.
She May Be Naked
Pirate's Log
bittersweet
Steel City Rust
Rants from the Pants
a duchess nonethelesss
Cast Iron Songs
Sonology
sweeping the kitchen
FOLK IS NOT HAPPY
Friday Night Dream
SEM REGRAS
Out of my Mind
adrianoblog
Who needs a TV?
MarkFisher's-MusicReview
From Brighton
Lost in the Shuffle
Stories From the City, Stories From the Sea
fractalpress.gr
Bleak Bliss
INDIEohren.!
LOST PLACES
Paulina Gamero. Media Studies A2
Chantelle Swain A2 Media Studies
Advanced Portfolio - Josh Pamfilo - Candidate Number 2186 - Centre Number 16607 - A2 Media Studies
The Themes of My Life
One Stunning Single Egg
If You Give a Girl a Camera...
Angie Dynamo
the traveling neighborhood
Happy Accidents
GLI Press
STATUS

### Included Files:

blogdata.txt, makedend.py, dend.txt, dend.jpg

## 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:

In order to cluster the blogs using K-Means, I used the function "kcluster(data,k)" from the script provided by "PCI" textbook and saved the script to a file "kclust.py" to run it.

Listing 6: The content of kclust.py

```
from math import *
import random

def readfile(filename):
    lines=[line for line in file(filename)]

    # First line is the column titles
    colnames=lines[0].strip( ).split('\t')[1:]
    rownames=[]
    data=[]
    for line in lines[1:]:
        p=line.strip( ).split('\t')
        # First column in each row is the rowname
        rownames.append(p[0])
        # The data for this row is the remainder of the row
        data.append([float(x) for x in p[1:]])
    return rownames,colnames,data


def pearson(v1,v2):
    # Simple sums
    sum1=sum(v1)
    sum2=sum(v2)

    # Sums of the squares
    sum1Sq=sum([pow(v,2) for v in v1])
    sum2Sq=sum([pow(v,2) for v in v2])

    # Sum of the products
    pSum=sum([v1[i]*v2[i] for i in range(len(v1))])

    # Calculate r (Pearson score)
    num=pSum-(sum1*sum2/len(v1))
    den=sqrt((sum1Sq-pow(sum1,2)/len(v1))*(sum2Sq-pow(sum2,2)/len(v1)))
    if den==0: return 0

    return 1.0-num/den
```

```
def kcluster(rows,distance=pearson,k=4):
    # Determine the minimum and maximum values for each point
    ranges=[(min([row[i] for row in rows]),max([row[i] for row in rows]))
    for i in range(len(rows[0]))]

    # Create k randomly placed centroids
    clusters=[[random.random( )*(ranges[i][1]-ranges[i][0])+ranges[i][0]
    for i in range(len(rows[0]))] for j in range(k)]

    lastmatches=None
    for t in range(100):
        print ('Iteration %d' % t)
        bestmatches=[[] for i in range(k)]

        # Find which centroid is the closest for each row
        for j in range(len(rows)):
            row=rows[j]
            bestmatch=0
            for i in range(k):
                d=distance(clusters[i],row)
                if d<distance(clusters[bestmatch],row): bestmatch=i
            bestmatches[bestmatch].append(j)

        # If the results are the same as last time, this is complete
        if bestmatches==lastmatches: break
        lastmatches=bestmatches

        # Move the centroids to the average of their members
        for i in range(k):
            avgs=[0.0]*len(rows[0])
            if len(bestmatches[i])>0:
                for rowid in bestmatches[i]:
                    for m in range(len(rows[rowid])):
                        avgs[m]+=rows[rowid][m]
                for j in range(len(avgs)):
                    avgs[j]/=len(bestmatches[i])
                clusters[i]=avgs

    return bestmatches


blognames,words,data=readfile('blogdata.txt')

print 'For k = 5:\n'
print '----------\n'
kclust=kcluster(data,k=5)
for i in range(5):
    print ([blognames[r] for r in kclust[i]])

print '\n++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n'

print ('For k = 10:\n')
kclust=kcluster(data,k=10)
for i in range(10):
    print ([blognames[r] for r in kclust[i]])
print '\n+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n'

print ('For k = 20:')
kclust=kcluster(data,k=20)
for i in range(20):
    print ([blognames[r] for r in kclust[i]])

print '\n+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n'
```

The script clusters the blogs using K-Means for K = 5, 10, 20 respectively, and prints out all centroids. It also prints out the number of iterations that were required for K = 5, 10, 20. These values are:

For K = 5: 5 Iterations

For K = 10: 4 Iterations

For K = 20: 5 Iterations

The output is saved to a file "kclust.txt".

Listing 7: Running kclust.py

```
root@ima-app:/var/www/Hussam/A8# python kclust.py > kclust.txt
root@ima-app:/var/www/Hussam/A8# cat kclust.txt
For k = 5:

----------

Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
['Floorshime Zipper Boots', 'DaveCromwell Writes', 'Chantelle Swain A2 Media
    Studies', 'A2 MEDIA COURSEWORK JOINT BLOG', 'Paulina Gamero. Media Studies A2
    ', 'The Themes of My Life', 'Jasmine Hodge', 'Advanced Portfolio - Josh
    Pamfilo - Candidate Number 2186 - Centre Number 16607 - A2 Media Studies', '
    Myopiamuse', 'Revolver USA Distribution & Midheaven mailorder']
["Riley Haas' blog", 'Cuz Music Rocks', 'She May Be Naked', 'Pithy Title Here', '
    THE HUB', 'Steel City Rust', 'ORGANMYTH', 'GLI Press', 'Stephanie Veto
    Photography', 'The Great Adventure 2016', 'Stonehill Sketchbook', 'a duchess
    nonethelesss', 'jaaackie.', 'nonsense a la mode', 'Happy Accidents', 'music
    of the moment', 'Angie Dynamo', 'Bonjour Girl', 'Playing Favorites', "Pirate'
    s Log", 'Eli Jace | The Mind Is A Terrible Thing To Paste', 'My Name Is Blue
    Canary', "i'm in too truthful a mood", 'Beyond the pond', 'Mile In Mine', '
    The Ideal Copy', 'A layer of chips', 'from a voice plantation', 'Sonology', '
    Tremagazine', 'If You Give a Girl a Camera...', 'bittersweet', 'sweeping the
    kitchen', 'A Day in the Life of...Me!!', 'Rants from the Pants', 'STATUS', '
    Cherry Area', 'The Campus Buzz on WSOU', 'Hip In Detroit', "It'll Glow On You
    ", "She's mad but she's magic. There's no lie in her fire.", 'Make Up, Music
    & Fashion', 'Cast Iron Songs', 'simone goes', 'guardtheguardians']
['Spinitron Charts', "MarkEOrtega's Journalism Portfolio", 'IoTube :)', 'Boggle
    Me Thursday', 'One Stunning Single Egg', "MarkFisher's-MusicReview"]
['holaOLA', 'Did Not Chart', 'FOLK IS NOT HAPPY', '@65 Sounding Booth', 'GYPSY
    RHAPSODY', 'MAGGOT CAVIAR', 'Music-Drop Magazine', 'MTJR RANTS & RAVES ON
    MUSIC', '~ mavaffantastico ~', '*Sixeyes: by Alan Williamson', 'Everything
    Starts With an A...', 'Some Call It Noise....', 'KiDCHAIR', 'The Music Binge
    ', 'The Stark Online', 'I/LOVE/TOTAL/DESTRUCTION', 'F-Measure', 'Encore']
```

```
['Bleak Bliss', 'SEM REGRAS', 'Friday Night Dream', 'Web Science and Digital
    Libraries Research Group', 'Fran Brighton', 'Stories From the City, Stories
    From the Sea', 'Lost in the Shuffle', 'adrianoblog', 'fractalpress.gr', '
    INDIEohren.!', 'the traveling neighborhood', 'Who needs a TV?', 'macthemost',
     'Out of my Mind', 'LOST PLACES']

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

For k = 10:

Iteration 0
Iteration 1
Iteration 2
Iteration 3
['Floorshime Zipper Boots', 'DaveCromwell Writes', 'A2 MEDIA COURSEWORK JOINT
    BLOG', 'Advanced Portfolio - Josh Pamfilo - Candidate Number 2186 - Centre
    Number 16607 - A2 Media Studies']
['THE HUB', 'Stories From the City, Stories From the Sea', 'Lost in the Shuffle',
     'music of the moment', 'Playing Favorites', 'My Name Is Blue Canary', '
    Tremagazine', 'Myopiamuse']
['holaOLA', 'Did Not Chart', 'FOLK IS NOT HAPPY', 'INDIEohren.!', 'GYPSY RHAPSODY
    ', 'Everything Starts With an A...', 'A layer of chips', 'Some Call It Noise
    ....', 'The Music Binge', 'Jasmine Hodge', 'The Stark Online', 'Encore']
['Spinitron Charts', "MarkEOrtega's Journalism Portfolio", 'Eli Jace | The Mind
    Is A Terrible Thing To Paste', 'If You Give a Girl a Camera...', 'The Campus
    Buzz on WSOU']
['IoTube    :)', 'fractalpress.gr', 'MAGGOT CAVIAR', 'MTJR RANTS & RAVES ON MUSIC
    ', '~ mavaffantastico ~', '*Sixeyes: by Alan Williamson', 'F-Measure', "
    MarkFisher's-MusicReview"]
['SEM REGRAS', 'Friday Night Dream', 'adrianoblog', 'Paulina Gamero. Media
    Studies A2', 'Who needs a TV?', 'Out of my Mind']
['Bleak Bliss', 'Pithy Title Here', 'Steel City Rust', 'ORGANMYTH', 'Stonehill
    Sketchbook', 'a duchess nonethelesss', 'nonsense a la mode', 'Music-Drop
    Magazine', "i'm in too truthful a mood", 'Beyond the pond', 'macthemost', '
    Mile In Mine', 'KiDCHAIR', 'LOST PLACES', 'sweeping the kitchen', 'One
    Stunning Single Egg', "She's mad but she's magic. There's no lie in her fire.
    "]
['Web Science and Digital Libraries Research Group', 'Fran Brighton', 'Angie
    Dynamo', 'Bonjour Girl', "Pirate's Log", 'Boggle Me Thursday', 'The Ideal
    Copy', 'I/LOVE/TOTAL/DESTRUCTION', "It'll Glow On You", 'Cast Iron Songs']
["Riley Haas' blog", 'Cuz Music Rocks', 'She May Be Naked', 'GLI Press', '
    Stephanie Veto Photography', 'Chantelle Swain A2 Media Studies', 'jaaackie.',
     'Happy Accidents', '@65 Sounding Booth', 'from a voice plantation', '
    bittersweet', 'A Day in the Life of...Me!!', 'Rants from the Pants', 'STATUS
    ', 'Cherry Area', 'Make Up, Music & Fashion', 'simone goes']
['The Great Adventure 2016', 'the traveling neighborhood', 'The Themes of My Life
    ', 'Sonology', 'Hip In Detroit', 'Revolver USA Distribution & Midheaven
    mailorder', 'guardtheguardians']

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

For k = 20:
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
['ORGANMYTH', 'The Great Adventure 2016', 'A2 MEDIA COURSEWORK JOINT BLOG', '
    Music-Drop Magazine', 'MTJR RANTS & RAVES ON MUSIC', 'macthemost', '
    Myopiamuse']
['holaOLA', 'FOLK IS NOT HAPPY', '@65 Sounding Booth', 'fractalpress.gr', '~
    mavaffantastico ~', 'Everything Starts With an A...', "MarkFisher's-
    MusicReview"]
['The Ideal Copy', 'The Themes of My Life', 'KiDCHAIR', 'Advanced Portfolio -
    Josh Pamfilo - Candidate Number 2186 - Centre Number 16607 - A2 Media Studies
    ']
[]
['SEM REGRAS', 'Friday Night Dream', 'adrianoblog', 'Who needs a TV?', 'Out of my
     Mind']
['Happy Accidents', 'A layer of chips']
['music of the moment', 'Bonjour Girl', 'Playing Favorites', 'Tremagazine']
['INDIEohren.!', 'LOST PLACES']
['Floorshime Zipper Boots', 'DaveCromwell Writes', 'GYPSY RHAPSODY']
['Sonology', 'sweeping the kitchen', 'The Music Binge', 'Jasmine Hodge']
['Chantelle Swain A2 Media Studies', 'Boggle Me Thursday']
['Paulina Gamero. Media Studies A2']
['Cast Iron Songs']
['A Day in the Life of...Me!!', 'Make Up, Music & Fashion']
['Bleak Bliss', 'Fran Brighton', 'Stories From the City, Stories From the Sea', '
    Lost in the Shuffle', 'IoTube :)', 'One Stunning Single Egg']
['MAGGOT CAVIAR', 'Some Call It Noise....', 'I/LOVE/TOTAL/DESTRUCTION', 'F-
    Measure', 'Encore', 'Revolver USA Distribution & Midheaven mailorder', '
    guardtheguardians']
['Angie Dynamo', 'the traveling neighborhood']
['Cuz Music Rocks', 'Pithy Title Here', 'Steel City Rust', 'GLI Press', '
    Stephanie Veto Photography', 'Stonehill Sketchbook', 'a duchess nonethelesss
    ', 'jaaackie.', 'nonsense a la mode', "Pirate's Log", "i'm in too truthful a
    mood", 'Beyond the pond', 'Mile In Mine', 'bittersweet', 'Rants from the
    Pants', 'STATUS', "She's mad but she's magic. There's no lie in her fire.", '
    simone goes']
['THE HUB', "MarkEOrtega's Journalism Portfolio", 'My Name Is Blue Canary', 'from
     a voice plantation', 'If You Give a Girl a Camera...', 'The Campus Buzz on
    WSOU', 'Hip In Detroit', "It'll Glow On You"]
["Riley Haas' blog", 'She May Be Naked', 'Spinitron Charts', 'Web Science and
    Digital Libraries Research Group', 'Did Not Chart', 'Eli Jace | The Mind Is A
     Terrible Thing To Paste', '*Sixeyes: by Alan Williamson', 'The Stark Online
    ', 'Cherry Area']

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

root@ima-app:/var/www/Hussam/A8#
```

### Included Files:

blogdata.txt, kclust.py, kclust.txt

## 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:

In order to create the “blogs’ JPEG” using Multidimensional Scaling, I used the function “scaledown(data)” and “draw2d(coords,blognames,jpeg=’2d.jpg’)” from the script provided by “PCI” textbook and saved the script to a file “make2d.py” to run it.

Listing 8: The content of make2d.py

```
from math import *
import sys, random
from PIL import Image,ImageDraw

def readfile(filename):
    lines=[line for line in file(filename)]
    # First line is the column titles
    colnames=lines[0].strip( ).split('\t')[1:]
    rownames=[]
    data=[]
    for line in lines[1:]:
        p=line.strip( ).split('\t')
        # First column in each row is the rowname
        rownames.append(p[0])
        # The data for this row is the remainder of the row
        data.append([float(x) for x in p[1:]])
    return rownames,colnames,data

def getheight(clust):
  # Is this an endpoint? Then the height is just 1
  if clust.left==None and clust.right==None: return 1

  # Otherwise the height is the same of the heights of
  # each branch
  return getheight(clust.left)+getheight(clust.right)

def getdepth(clust):
  # The distance of an endpoint is 0.0
  if clust.left==None and clust.right==None: return 0

  # The distance of a branch is the greater of its two sides
  # plus its own distance
  return max(getdepth(clust.left),getdepth(clust.right))+clust.distance

def drawnode(draw,clust,x,y,scaling,labels):
  if clust.id<0:
    h1=getheight(clust.left)*20
    h2=getheight(clust.right)*20
    top=y-(h1+h2)/2
    bottom=y+(h1+h2)/2
    # Line length
    ll=clust.distance*scaling
    # Vertical line from this cluster to children
    draw.line((x,top+h1/2,x,bottom-h2/2),fill=(255,0,0))

    # Horizontal line to left item
    draw.line((x,top+h1/2,x+ll,top+h1/2),fill=(255,0,0))
    # Horizontal line to right item
    draw.line((x,bottom-h2/2,x+ll,bottom-h2/2),fill=(255,0,0))

    # Call the function to draw the left and right nodes
    drawnode(draw,clust.left,x+ll,top+h1/2,scaling,labels)
    drawnode(draw,clust.right,x+ll,bottom-h2/2,scaling,labels)
  else:
    # If this is an endpoint, draw the item label
    draw.text((x+5,y-7),labels[clust.id],(0,0,0))

def tanamoto(v1,v2):
  c1,c2,shr=0,0,0

  for i in range(len(v1)):
    if v1[i]!=0: c1+=1 # in v1
    if v2[i]!=0: c2+=1 # in v2
    if v1[i]!=0 and v2[i]!=0: shr+=1 # in both

  return 1.0-(float(shr)/(c1+c2-shr))


def pearson(v1,v2):
    # Simple sums
    sum1=sum(v1)
    sum2=sum(v2)

    # Sums of the squares
    sum1Sq=sum([pow(v,2) for v in v1])
    sum2Sq=sum([pow(v,2) for v in v2])

    # Sum of the products
    pSum=sum([v1[i]*v2[i] for i in range(len(v1))])

    # Calculate r (Pearson score)
    num=pSum-(sum1*sum2/len(v1))
    den=sqrt((sum1Sq-pow(sum1,2)/len(v1))*(sum2Sq-pow(sum2,2)/len(v1)))
    if den==0: return 0

    return 1.0-num/den


def scaledown(data,distance=pearson,rate=0.01):
  n=len(data)

  # The real distances between every pair of items
  realdist=[[distance(data[i],data[j]) for j in range(n)]
            for i in range(0,n)]

  # Randomly initialize the starting points of the locations in 2D
  loc=[[random.random(),random.random()] for i in range(n)]
  fakedist=[[0.0 for j in range(n)] for i in range(n)]

  lasterror=None
  for m in range(0,1000):
    # Find projected distances
    for i in range(n):
      for j in range(n):
        fakedist[i][j]=sqrt(sum([pow(loc[i][x]-loc[j][x],2)
                                 for x in range(len(loc[i]))]))

    # Move points
    grad=[[0.0,0.0] for i in range(n)]

    totalerror=0
    counter = m+1
    for k in range(n):
      for j in range(n):
        if j==k: continue
        # The error is percent difference between the distances
        if (realdist[j][k] <> 0):
          errorterm=(fakedist[j][k]-realdist[j][k])/realdist[j][k]

        # Each point needs to be moved away from or towards the other
        # point in proportion to how much error it has
        grad[k][0]+=((loc[k][0]-loc[j][0])/fakedist[j][k])*errorterm
        grad[k][1]+=((loc[k][1]-loc[j][1])/fakedist[j][k])*errorterm

        # Keep track of the total error
        totalerror+=abs(errorterm)
    print counter, ' :', totalerror

    # If the answer got worse by moving the points, we are done
    if lasterror and lasterror<totalerror: break
    lasterror=totalerror

    # Move each of the points by the learning rate times the gradient
    for k in range(n):
      loc[k][0]-=rate*grad[k][0]
      loc[k][1]-=rate*grad[k][1]

  return loc


def draw2d(data,labels,jpeg='mds2d.jpg'):
    img=Image.new('RGB',(2000,2000),(255,255,255))
    draw=ImageDraw.Draw(img)
    for i in range(len(data)):
        x=(data[i][0]+0.5)*1000
        y=(data[i][1]+0.5)*1000
        draw.text((x,y),labels[i],(0,0,0))
    img.save(jpeg,'JPEG')

blognames,words,data=readfile('blogdata.txt')
coords=scaledown(data)
draw2d(coords,blognames,jpeg='2d.jpg')
```

The number of iterations that was required to go from 4176.44795059 down to 2713.69160666 average error, before the error began to increase, is 238 iterations.

**Note:** I ran the script multiple times, and the number of iterations required was largely different from previous runs of the same script.

The text output is saved to a file “kclust.txt”.

The resulted JPEG file is named “2d.jpg”.

Listing 9: Running make2d.py

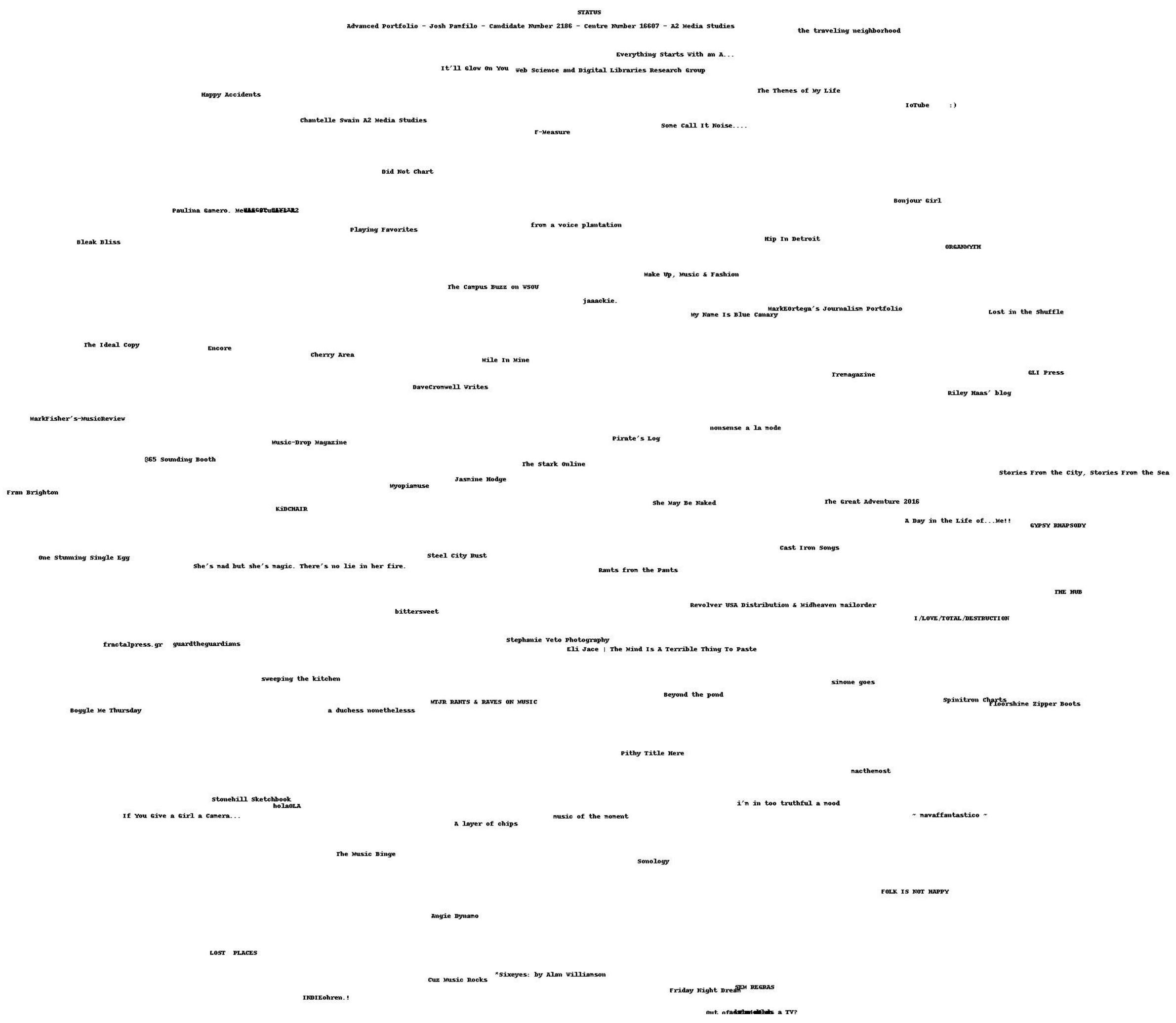
```
root@ima-app:/var/www/Hussam/A8# python make2d.py > 2d.txt
root@ima-app:/var/www/Hussam/A8# cat 2d.txt
1  : 4176.44795059
2  : 3199.28206864
3  : 3087.63197799
4  : 3035.86849153
5  : 3000.05251741
6  : 2974.06006622
7  : 2950.81068454
8  : 2931.81928236
9  : 2913.52420971
10  : 2897.28266551
11  : 2883.94216807
12  : 2874.55698459
13  : 2866.14284263
14  : 2859.09522785
15  : 2852.51996715
16  : 2847.93589556
17  : 2844.97611319
18  : 2843.37581751
19  : 2842.17059266
20  : 2841.16455409
21  : 2840.06541549
22  : 2839.05109437
23  : 2837.74763181
24  : 2836.43804045
25  : 2835.15695415
26  : 2833.87829561
27  : 2832.80149691
28  : 2831.67786657
29  : 2830.65511729
30  : 2829.87435441
31  : 2829.32153715
32  : 2828.55376826
33  : 2827.7496543
34  : 2826.86344611
35  : 2825.74841949
36  : 2824.46203448
37  : 2823.2026712
38  : 2822.01411055
39  : 2820.80002279
40  : 2819.48607499
41  : 2818.18110124
42  : 2816.86419709
43  : 2815.44030833
44  : 2814.0899552
45  : 2812.95682783
46  : 2811.89186831
47  : 2810.89183676
48  : 2810.01054415
49  : 2809.2273762
50  : 2808.50086884
```

```
51  : 2807.62861644
52  : 2806.66489255
53  : 2805.63905059
54  : 2804.56841114
55  : 2803.43278769
56  : 2802.28649074
57  : 2801.14982711
58  : 2799.95583679
59  : 2798.82267525
60  : 2797.70255201
61  : 2796.56387824
62  : 2795.46240635
63  : 2794.44451683
64  : 2793.24758655
65  : 2791.99860426
66  : 2790.80947367
67  : 2789.49693255
68  : 2788.28645392
69  : 2787.15717997
70  : 2786.23556888
71  : 2785.3941433
72  : 2784.53589236
73  : 2783.64837053
74  : 2782.75089986
75  : 2781.85139077
76  : 2781.04046432
77  : 2780.23940719
78  : 2779.42855537
79  : 2778.56931082
80  : 2777.64189775
81  : 2776.61097452
82  : 2775.54424706
83  : 2774.59994577
84  : 2773.84374295
85  : 2772.9943342
86  : 2772.13348857
87  : 2771.26367692
88  : 2770.57112026
89  : 2769.9542129
90  : 2769.45393203
91  : 2769.08579919
92  : 2768.74806773
93  : 2768.44144758
94  : 2768.24060634
95  : 2768.0129232
96  : 2767.77405026
97  : 2767.60589926
98  : 2767.43850514
99  : 2767.26821362
100  : 2767.16499183
101  : 2767.08604513
102  : 2766.97426806
103  : 2766.76250708
104  : 2766.48725457
105  : 2766.20644509
106  : 2765.82132743
```

```
107  : 2765.39982485
108  : 2764.95634555
109  : 2764.46500771
110  : 2763.98319869
111  : 2763.48741703
112  : 2762.94099281
113  : 2762.34626919
114  : 2761.71975662
115  : 2761.05235516
116  : 2760.32365333
117  : 2759.49306678
118  : 2758.63609483
119  : 2757.69295584
120  : 2756.67548966
121  : 2755.59124786
122  : 2754.53280881
123  : 2753.61031153
124  : 2752.85993519
125  : 2752.1613236
126  : 2751.57403018
127  : 2750.99693198
128  : 2750.447924
129  : 2749.8400888
130  : 2749.1683332
131  : 2748.47100815
132  : 2747.82876622
133  : 2747.13431568
134  : 2746.4293871
135  : 2745.68410759
136  : 2744.89026099
137  : 2744.1631112
138  : 2743.4542112
139  : 2742.7088057
140  : 2741.95893598
141  : 2741.37354834
142  : 2740.85058511
143  : 2740.36835667
144  : 2739.84523429
145  : 2739.34624391
146  : 2738.94683379
147  : 2738.5541387
148  : 2738.10896558
149  : 2737.6453798
150  : 2737.20503215
151  : 2736.82617571
152  : 2736.46765133
153  : 2736.14620665
154  : 2735.81239356
155  : 2735.45724624
156  : 2735.13463109
157  : 2734.82977912
158  : 2734.51466014
159  : 2734.2250341
160  : 2733.95400037
161  : 2733.71627304
162  : 2733.50849413
```

```
163  : 2733.32717627
164  : 2733.15803307
165  : 2732.99852703
166  : 2732.8152737
167  : 2732.60819925
168  : 2732.39483262
169  : 2732.15388231
170  : 2731.90814637
171  : 2731.59832104
172  : 2731.24195043
173  : 2730.86158811
174  : 2730.41301182
175  : 2729.96852227
176  : 2729.50702729
177  : 2729.02699885
178  : 2728.46415847
179  : 2727.86506474
180  : 2727.41350813
181  : 2726.99116034
182  : 2726.5653195
183  : 2726.13225936
184  : 2725.72839041
185  : 2725.31053573
186  : 2724.86531614
187  : 2724.43720122
188  : 2724.11646089
189  : 2723.85376619
190  : 2723.62199869
191  : 2723.37974772
192  : 2723.11971778
193  : 2722.8132779
194  : 2722.53031244
195  : 2722.29335539
196  : 2722.10370126
197  : 2721.87285605
198  : 2721.5872956
199  : 2721.2471181
200  : 2720.89240326
201  : 2720.52725841
202  : 2720.18213744
203  : 2719.83511486
204  : 2719.55826524
205  : 2719.41760148
206  : 2719.27551294
207  : 2719.14204388
208  : 2718.95806795
209  : 2718.73566583
210  : 2718.47296645
211  : 2718.221627
212  : 2717.97024178
213  : 2717.69059412
214  : 2717.37282433
215  : 2717.04614113
216  : 2716.71438665
217  : 2716.38270864
218  : 2716.08187315
```

```
219  : 2715.83634292
220  : 2715.65838332
221  : 2715.51324516
222  : 2715.41054588
223  : 2715.29736663
224  : 2715.17143061
225  : 2715.03777637
226  : 2714.88833114
227  : 2714.73588278
228  : 2714.57570795
229  : 2714.45622897
230  : 2714.37205487
231  : 2714.26687479
232  : 2714.15655162
233  : 2714.03444989
234  : 2713.94059924
235  : 2713.85364815
236  : 2713.75560392
237  : 2713.69694917
238  : 2713.69160666
239  : 2713.69509067
root@ima-app:/var/www/Hussam/A8# ls
100blogs.txt 2d.txt      dend.jpg generatefeedvector.py getpages.py kclust.txt
    makedend.py pages.txt     urls.txt
2d.jpg       blogdata.txt dend.txt getblogurls.py      kclust.py   make2d.py
    makeunique.py uniqueurls.txt
root@ima-app:/var/www/Hussam/A8#
```



## Included Files:

blogdata.txt, make2d.py, 2d.txt, 2d.jpg