Part 1

Wrote script getblogs.pt to create a list of 100 blogs in a text file, then manually added the two required blogs, and used notepad++ to remove “?expref=next-blog” from the end of the saved urls.

Using the urls saved to bloglist.txt, modified provided code in generatefeedvector.py to create the required matrix. Modified code was saved as blogmatrix.py. Code was originally modified to use only words found in the blog titles, but the resulting matrix was far too small, so words from the blog summaries were re-included. To limit the size of the word list to 1000 terms, the list is cut, removing terms past the 1000th. I believe this will be sufficient, as any term found and added that late into the list being formed is not going to be important enough to miss.

Part 2

First, had to adjust clusters.py to run in python3, which only needed correction to file input and print lines. After corrections to clusters.py, used script provided in week 11 slide 13 in a file blogdendo.py to create the required ascii and jpg dendrograms. Ascii dendrogram is saved as asciidendro.txt. Jpg dendrogram is saved as the default, clusters.jpg.

Part 3

Created a separate file kclusters.py for this part. Script was simple, reusing the readfile line from blogdendo.py. Output was directed into a text file kclusters.txt. Output in the file indicates that k values of 5 and 10 required 4 iterations, while the k value 20 required one less, at three iterations.

Part 4

For this final part, once again created a separate file makeMDS.py to create the required .jpg, using lines of code provided as examples in slide 29. Standard output was piped into MDSout.txt, while the resulting .jpg was saved as MDS.jpg.

Code

#getblogs.py

**import** urllib**.**request

**from** time **import** sleep

x **=** 0

**for** x **in** range **(**0**,** 100**):**

**try:**

url **=** 'https://www.blogger.com/next-blog?navBar=true&blogID=953024975153422094'

response **=** urllib**.**request**.**urlopen**(**url**)**

#print (response.text)

**with** open**(**"Output.txt"**,** "a"**)** **as** text\_file**:**

**print(**response**.**geturl**(),** file**=**text\_file**)**

sleep**(**2**)**

**except:**

x **=** x **-** 1

**pass**

#blogmatrix.py

#limit to 1000 terms

**import** sys

**import** feedparser

**import** re

# Returns title and dictionary of word counts for an RSS feed

**def** getwordcounts**(**url**):**

# Parse the feed

d**=**feedparser**.**parse**(**url**)**

wc**={}**

# Loop over all the entries

**for** e **in** d**.**entries**:**

**if** 'summary' **in** e**:** summary**=**e**.**summary

**else:** summary**=**e**.**description

# Extract a list of words

words**=**getwords**(**e**.**title**+**' '**+**summary**)**

**for** word **in** words**:**

wc**.**setdefault**(**word**,**0**)**

wc**[**word**]+=**1

**return** d**.**feed**.**title**,**wc

**def** getwords**(**html**):**

# Remove all the HTML tags

txt**=**re**.**compile**(**r'<[^>]+>'**).**sub**(**''**,**html**)**

# Split words by all non-alpha characters

words**=**re**.**compile**(**r'[^A-Z^a-z]+'**).**split**(**txt**)**

# Convert to lowercase

**return** **[**word**.**lower**()** **for** word **in** words **if** word**!=**''**]**

#test for getting wordcounts

#title,wc = getwordcounts('http://f-measure.blogspot.com/feeds/posts/default')

#print(title)

#print(wc)

apcount**={}**

wordcounts**={}**

infile **=** open**(**'bloglist.txt'**)**

**for** line **in** infile**:**

**try:**

line **=** line**.**strip**()**

feed **=** line **+** 'feeds/posts/default'

title**,**wc**=**getwordcounts**(**feed**)**

wordcounts**[**title**]=**wc

**for** word**,**count **in** wc**.**items**():**

apcount**.**setdefault**(**word**,**0**)**

**if** count**>**1**:**

apcount**[**word**]+=**1

**except:**

**print** **(**'Failed to parse feed %s' **%** feed**)**

wordlist**=[]**

**for** w**,**bc **in** apcount**.**items**():**

frac**=**float**(**bc**)/**len**(**feed**)**

**if** frac**>**0.1 **and** frac**<**0.5**:**

wordlist**.**append**(**w**)**

out**=**open**(**'blogdata1.txt'**,**'w'**)**

out**.**write**(**'Blog'**)**

wordlist **=** wordlist**[:**1000**]** # reduce to 1000 entries if greater than

**for** word **in** wordlist**:** out**.**write**(**'\t%s' **%** word**)**

out**.**write**(**'\n'**)**

**for** blog**,**wc **in** wordcounts**.**items**():**

**print** **(**blog**)**

out**.**write**(**blog**)**

**for** word **in** wordlist**:**

**if** word **in** wc**:** out**.**write**(**'\t%d' **%** wc**[**word**])**

**else:** out**.**write**(**'\t0'**)**

out**.**write**(**'\n'**)**

**print** **(**str**(**len**(**wordlist**)))** #check on wordlist length

#blogdendo.py

**import** clusters

**import** sys

#out=open('blogdendo1.txt','w')

#drawdendrogram(clust, labels, jpeg='clusters.jpg') // img.save(jpeg, 'JPEG')

#readfile(filename) //return (rownames, colnames, data)

#kcluster(rows, distance=pearson, k=4) //return bestmatches

blognames**,**words**,**data**=**clusters**.**readfile**(**'blogdata1.txt'**)** # returns blog titles, words in blog (10%-50% boundaries), list of frequency info

clust**=**clusters**.**hcluster**(**data**)** # returns a tree of foo.id, foo.left, foo.right

sys**.**stdout **=** open**(**'asciidendo.txt'**,** 'w'**)**

clusters**.**printclust**(**clust**,**labels**=**blognames**)** # walks tree and prints ascii approximation of a dendogram; distance measure is Pearson's r

clusters**.**drawdendrogram**(**clust**,**blognames**)**

#kclusters.py

**import** clusters

**import** sys

blognames**,**words**,**data**=**clusters**.**readfile**(**'blogdata1.txt'**)** # returns blog titles, words in blog (10%-50% boundaries), list of frequency info

clust**=**clusters**.**hcluster**(**data**)** # returns a tree of foo.id, foo.left, foo.right

sys**.**stdout **=** open**(**'kclusters.txt'**,** 'w'**)**

**print(**'k=5'**)**

kclust**=**clusters**.**kcluster**(**data**,**k**=**5**)**

**print(**''**)**

**print(**'k=10'**)**

kclust**=**clusters**.**kcluster**(**data**,**k**=**10**)**

**print(**''**)**

**print(**'k=20'**)**

kclust**=**clusters**.**kcluster**(**data**,**k**=**20**)**

#makeMDS.py

**import** clusters

**import** sys

blognames**,**words**,**data**=**clusters**.**readfile**(**'blogdata1.txt'**)** # returns blog titles, words in blog (10%-50% boundaries), list of frequency info

sys**.**stdout **=** open**(**'MDSout.txt'**,** 'w'**)**

coords**=**clusters**.**scaledown**(**data**)**

clusters**.**draw2d**(**coords**,**blognames**,**jpeg**=**'MDS.jpg'**)**

List of URIs used:

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

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

http://ohsustudent.blogspot.com/

https://vcorejava.blogspot.com/

http://1place.blogspot.com/

http://futureberhampur.blogspot.com/

http://uccomputinghistory.blogspot.com/

http://rmshark.blogspot.com/

http://socarchsci.blogspot.com/

https://chemistry-mac.blogspot.com/

http://www.alwaysavoidalliteration.com/

http://leadershipandhardiness.blogspot.com/

https://iesmcrc-hrclub.blogspot.com/

http://yepd2016.blogspot.com/

http://midwoodblogs.blogspot.com/

http://restauro-del-libro.blogspot.com/

http://novembereverafter.blogspot.com/

https://nehrujarpula.blogspot.com/

http://adasistemas.blogspot.com/

http://moocobsessive.blogspot.com/

http://fivethingsthatdontsuck.blogspot.com/

http://annabelriley.blogspot.com/

https://fulbrightmalaysiaeta.blogspot.com/

http://greattide.blogspot.com/

http://workhardplayhardeatwell.blogspot.com/

http://project1stgen.blogspot.com/

http://www.douglasproctor.com/

http://lifeatsouthampton.blogspot.com/

http://nhatrangcitynews.blogspot.com/

http://spotlight.uis.edu/

https://cpcs-int.blogspot.com/

http://blackcollege.blogspot.com/

http://richardjeffries.blogspot.com/

http://earlymodern-lit.blogspot.com/

http://www.pramodyadav.com.np/

http://maheshdholiya.blogspot.com/

https://plu-system-notices.blogspot.com/

http://www.margarethageertsemasligh.com/

http://allthingsniceandgingerful.blogspot.com/

http://shengshawnhillaryentertainment.blogspot.com/

http://gustavetheitaliannoodle.blogspot.com/

http://comtechnews.blogspot.com/

http://ushistorycurrentevents.blogspot.com/

http://www.dbntrust.org/

http://stievie.blogspot.com/

http://nraoemkc.blogspot.com/

http://contemporarycapture.blogspot.com/

http://androancap.blogspot.com/

http://baphotoboltonuniversity.blogspot.com/

http://sigcse2008.blogspot.com/

http://philiphall42.blogspot.com/

http://hmloveland.blogspot.com/

http://anglosaxonnorseandceltic.blogspot.com/

http://sbradiosports.blogspot.com/

http://blog.campusbuddy.com/

http://anandwadadekar.blogspot.com/

http://convergencefa2016.blogspot.com/

http://caplawopd.blogspot.com/

http://longhornfin.blogspot.com/

http://placeforthestolen.blogspot.com/

http://thisismyblogspotpage.blogspot.com/

http://teachingbconference.blogspot.com/

http://collettitz.blogspot.com/

http://artwithmeyers.blogspot.com/

http://ajcetbi.blogspot.com/

http://sjsuartlibrarian.blogspot.com/

http://zusvendetta.blogspot.com/

http://cowboysgalsblog.blogspot.com/

https://biotechnologistpandit.blogspot.com/

http://jjmcqueen.blogspot.com/

http://hoostories.blogspot.com/

http://bayareafinnishnetwork.blogspot.com/

https://devcomnigeria.blogspot.com/

http://livinginthelibraryworld.blogspot.com/

http://uofllawschool.blogspot.com/

http://dpiwis-tepdl.blogspot.com/

http://siuctechdev.blogspot.com/

http://cseallinone.blogspot.com/

http://announcingwestwood.blogspot.com/

http://jooanddaniel.blogspot.com/

http://makangale.blogspot.com/

http://lovefrommarbella.blogspot.com/

http://lzsisters.blogspot.com/

http://nmsueddyag.blogspot.com/

http://theredpenmumbai.blogspot.com/

http://heatherha.blogspot.com/

https://4luckylove.blogspot.com/

http://geographywithdan.blogspot.com/

http://aapabandit.blogspot.com/

http://dearnoah.blogspot.com/

http://jhrminutes.blogspot.com/

http://philosophysmoker.blogspot.com/

http://blog.campusbuddy.com/

http://ushistorycurrentevents.blogspot.com/

http://www.specialmata.com.ng/

http://joeltrain.blogspot.com/

http://www.specialmata.com.ng/

http://boris-denisov.blogspot.com/

http://www.wisconsinhistoricalmarkers.com/p/home-page.html

https://chemistry-mac.blogspot.com/