Assignment 8

CS532-s16: Web Sciences

Spring 2016

John Berlin

Generated on April 6, 2016

Index

Question 1

```
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) 500 terms,
this is *after* the criteria on p. 32 (slide 7) has been satisfied.
```

Answer

In getting the 100 blogs and generating the blog-term matrix I wrote two python scripts *getFeeds.py* and *processData.py*. The first script getFeeds.py which is seen in listing 2 does the job of getting the blog feed data. Then processData.py seen in listing 3 does the generation of the blog-term matrix file plus generation of following questions data.

Getting the feed data turned out to require a little bit more effort than I was expecting. The extra effort came into play when I came across blogs with no titles(blog titles). These no blog title blogs as a majority had little to no text as I found out when a great deal of my words were blanks. Also adding to the extra effort was realizing that the random next blog was not truly random rather pseudo-random as I had duplicates. Besides that the retrieval of the blog data was rather straightforward.

The blogspot API was rather straightforward and required few calls. The uris used to interact with the blogspot API can be seen on lines 12,13,14, and 15 in listing 2. To speed up the process of getting the feed entries I appended *?max-results=200* to end of the feeds uri(*feeds/posts/default*). By doing this I get a maximum of 200 entries per get request, which greatly limits the number of API calls required to get paginated entries for blogs with over 200 posts.

I will explain the code seen in listing 2 at first generally then break down the methods used to accomplish the task. As always consult the code seen in listing 2 for more details in the comments.

1. Entire process(getData)
   (a) Check to see if the data file directory exists if not create it
   (b) Set up session and add user agent so we appear less like a robot
   (c) Consume and process the WebScience and Digital Libraries blog posts
   (d) Consume and process F-Measure blog posts
   (e) Ask for next blog 98 times, consume and process its posts
   (f) Write blog urls to file and write blog terms to json file called blog-data.json
2. get98
   (a) While count is less than 98
   (b) Get the next blog
   (c) Check to see if we have already gotten it if we have choose another otherwise consume_all
   (d) If what we got is nothing choose another
   (e) Add what we got to the data collection, increment counter by one and choose another
3. consume_all(start, sesh)
   (a) Ask the api for the initial set of feed entries
   (b) Parse the feed and get the title, if the blog has no title or if the number of entries is less than 25 choose another
   (c) Check to see if we have more feed entries
   (d) Process and flatten the initial set of feed entries
   (e) While we have more entries(pagination), ask for next set of feed entries, process and flatten, then check again
   (f) Do preliminary word count as the number of words if not reduces can cause the file size to be to big
   (g) Finally return the title and the blogs terms
4. process_text(text)
   (a) Process the feeds text using BeautifulSoup to clean out html, remove all non alpha characters then make the text lower case

(b) Use nltk(natural language toolkit) to tokenize the text into words, filter the words by checking to see if they are not English language stop words

(c) Return list of valid words

5. check_next(text)

(a) Use BeautifulSoup to process the entire feed

(b) Extract the next link(next set of feed entries) by looking for *link,type: application/atom+xml, rel: next*

(c) If that link exists the returned list will be of size 1, return the link and true otherwise false and none

The list of 98 urls gotten can be seen in listing 1.

Now I will also explain the other file processData.py seen in listing 3, here in its entirety for the parts that go with question one. In the remain question sections I will go into detail about what was required for the parts pertaining to those remaining questions. Please note that I used clusters.py file that accompanied the book found in github repo containing the example code.

For processing the data for the top 500 terms I created a class called feed. It allows for quick access to the preliminary word count and stem count. For another part for the next questions I stemmed the words to get a better clusters as the original method returned a very sparse term matrix leading to errors in distance calculations. Stemming the words I again used the natural language tool kit's(nltk) English stemmer to produce a correct stemming for the English language.

The two methods in processData.py listing 3 that partaine to question one are *generate_blogfile* and *generate_blogfile_stem*. Both execute as such

1. generate_blogfile

(a) Read in the blog term json file. I use the functional python library seq class to allow for short and concise expression of the processing.

(b) Get the top 500 by

(c) Flattening each blogs word count

(d) Filtering word count tuples for counts greater than 10(non-stemmed),1(stemmed)

(e) Do a total word count

(f) Filter by the faked tfidf function

(g) Sort the data in descending order

(h) Take the top 500

(i) Transform the data to words only and make it a list

(j) Sort the top 500 words alphabetically

(k) Write the words out the blog-term matrix file

2. generate_blogfile_stem

   (a) Behaves exactly like generate_blogfile except use the stemmed word count and gets words with a count greater than 1(addendum to value used in 1d)

I mentioned that words with intra-document counts of 10 were used for the non-stemmed version in the explanation of 1d. Before using those values, the default count of 1 was used which lead to divide by zero exceptions in the MDS calculation. On inspection I believe it was caused by the sparse nature of the matrix terms overall. This change was done well after I had written the code for the initial implementation which lead to my usage of stemmed words. Also with this change I was able to remove my changes to the original clusters.py file which checked and poorly corrected the divide by zero exception.

```
http://deadbeatfanzine.blogspot.com
http://thebeautifultrashart.blogspot.com
http://hiiijaaackie.blogspot.com
http://encorenorthernireland.blogspot.com
http://blog.spinitron.com
http://floorshimezipperboots.blogspot.com
http://lafotografiaefectistaabstracta.blogspot.com
http://pithytitlehere.blogspot.com
http://thesportsmith.blogspot.com
http://ahtapotunbahcesi.blogspot.com
http://dancingincirclesnow.blogspot.com
http://rosiegigga2media.blogspot.com
http://thekidsarecoming.blogspot.com
http://www.radioshower.com
http://desolationrowrecords.blogspot.com
http://isyelili.blogspot.com
http://theindiefriend.blogspot.com
http://parishradio.blogspot.com
http://listeningear.blogspot.com
http://stonehillsketchbook.blogspot.com
http://mattsbunker.blogspot.com
http://www.thejeopardyofcontentment.com
http://theworldsfirstinternetbaby.blogspot.com
http://sixtyat60.blogspot.com
http://jamiemclelland.blogspot.com
http://www.samtasticreview.com
http://kidchair.blogspot.com
http://bogglemethursday.blogspot.com
http://markeortega.blogspot.com
http://ihatethe90s.blogspot.com
http://storiesfromthecityradiovalencia.blogspot.com
http://thefastbreakofchampions.blogspot.com
http://skinnyshoes.blogspot.com
http://semregrasluispink.blogspot.com
http://elijace.blogspot.com
http://marialombideezpeleta.blogspot.com
http://ilovetotaldestruction.blogspot.com
http://ngaio1619.blogspot.com
http://flatbasset.blogspot.com
http://hartsdesire.blogspot.com
http://sixeyes.blogspot.com
http://rantsfromthepants.blogspot.com
http://spicyseatdolphin.blogspot.com
http://my-name-is-blue-canary.blogspot.com
http://dana9morgan.blogspot.com
http://moontopples.blogspot.com
http://dinosaursarefun.blogspot.com
http://ashleyemwarren.blogspot.com
http://noradiorecs.blogspot.com
http://thehubkxci.blogspot.com
http://lostintheshuffle899.blogspot.com
http://flowradio.blogspot.com
http://mysteryfallsdown.blogspot.com
http://thenightmail.blogspot.com
http://seveninchesisenough.blogspot.com
http://didnotchart.blogspot.com
http://rodshone.blogspot.com
```

```
http://mileinmine.blogspot.com
http://mesastivromia.blogspot.com
http://theonionfield.blogspot.com
http://globalgoon.blogspot.com
http://beckysharpfashionblog.blogspot.com
http://ourpodcastcouldbeyourlife.blogspot.com
http://steel-city-rust.blogspot.com
http://maggotcaviar.blogspot.com
http://www.sonology.com
http://thetremagazine.blogspot.com
http://flipmpip.blogspot.com
http://psychfolkmusic.blogspot.com
http://kunstlertreu.blogspot.com
http://mondaywakeup.blogspot.com
http://thepowerofindependenttrucking.blogspot.com
http://richardwhitten.blogspot.com
http://marshwiggle.blogspot.com
http://dustandwaterstudios.blogspot.com
http://harisphotonet.blogspot.com
http://mo-forgetaboutit.blogspot.com
http://kistefm.blogspot.com
http://mobbie2.blogspot.com
http://hani-bittersweet.blogspot.com
http://jbreitling.blogspot.com
http://turnitupjack.blogspot.com
http://davecromwellwrites.blogspot.com
http://mtjrrantsravesonmusic.blogspot.com
http://ps-music.blogspot.com
http://superchicken46.blogspot.com
http://cherryarea.blogspot.com
http://jlmdlhlcm1516.blogspot.com
http://trembleunderboomlights.blogspot.com
http://onestunningsingleegg.blogspot.com
http://chantellesmedia2.blogspot.com
http://everydaymusicportland.blogspot.com
http://karldrinkwater.blogspot.com
http://jakobclapensteam.blogspot.com
http://adrianomarquesblog.blogspot.com
http://www.chrisanne-grise.com
http://campusbuzzwsou.blogspot.com
http://jojobethkatiehannahlcm1516.blogspot.com
```

Listing 1: The 98 uris

```
import json as jjson
import re
from operator import add
import os
import feedparser
import nltk
import requests
from bs4 import BeautifulSoup
from functional import seq
from nltk.corpus import stopwords

fmeasure = "http://f-measure.blogspot.com/feeds/posts/default?max-results=200"
fmeasure_next = "https://www.blogger.com/next-blog?navBar=true&blogID=3471633091411211117"
wsdl = "http://ws-dl.blogspot.com/feeds/posts/default?max-results=200"
bspot_feed = "/feeds/posts/default?max-results=200"

# nuke all none-alpha chars
removeExtra = re.compile('[^a-zA-Z]')

langs = ["dutch", "finnish", "german", "italian", "portuguese", "spanish", "turkish", "danish", "english",
         "french", "hungarian", "norwegian", "russian", "swedish"]
langStopWords = {}
for lang in langs:
    langStopWords[lang] = stopwords.words(lang)


def check_next(text):
    # check for the next button ie pagination of blog pages
    soup = BeautifulSoup(text, "lxml-xml")
    next_page = soup.find_all('link', attrs={'type': 'application/atom+xml', 'rel': 'next'})
    # if there is a next page our next_page list will always be 1 otherwise its 0
    # that means we have consumed all the pages for the blog
    if len(next_page) > 0:
        nl = next_page[0].attrs['href']
        return True, nl
    return False, None


def process_text(text):
    # clean text
    t = removeExtra.sub(' ', BeautifulSoup(text.summary, 'html5lib').text).lower()
    ret = []
    # tokenize according to word and emit word if it is not a stopword
    for word in nltk.word_tokenize(t):
        if word not in langStopWords['english']:
            ret.append(word)
    return ret


```

```
def consume_all(start, sesh):
    # the text of the entry titles
    text = []
    # the request for the first page of the blogs atom feed
    r = sesh.get(start)  # type: requests.Response
    # parse the feed
    feed = feedparser.parse(r.text)  # type: feedparser.
    FeedParserDict
    # some blogs do not have titles so I must check for them
    try:
        t = feed['feed']['title']
    except KeyError as e:
        # some blogs do not have a title so I will skip them
        print(e)
        r.close()
        return None
    # see if we have more feed pages
    good, nl = check_next(r.text)

    if len(feed.entries) < 25:
        return None
    # All these operations are very good for more of a functional
    approach
    # to much work otherwise
    '''
    flatmap:
        for entry <- feed, token <- process_text(entry): yeild
    token
    '''
    text.extend(seq(feed.entries).flat_map(process_text).to_list())
    while good:
        r = sesh.get(nl)
        feed = feedparser.parse(r.text)
        r.close()
        text.extend(seq(feed.entries).flat_map(process_text))
        good, nl = check_next(r.text)

    # since the number of words can become extremely large we need
    to do a
    # reduce before emitting the words
    text = seq(text).map(lambda word: (word, 1)) \
        .reduce_by_key(add) \
        .order_by(lambda x: x[1]).to_dict()

    return {'title': t, 'text': text}


def get98(sesh):
    the98 = {}
    counter = 0
    while counter < 98:
        r = sesh.get(fmeasure_next)  # type: requests.Response
        rurl = r.url[:r.url.rindex("/")]
        # I had the random button give the same blog 68 times
    before
        # so I will simply look for another one if I have seen it
    before
```

```
        if the98.get(rurl, None) is not None:
            r.close()
            print("Processed URL already %s" % rurl)
            continue
        else:
            print(r.url[:r.url.rindex("/")])
        got = consume_all(rurl + bspot_feed, sesh)
        # if the consuming of the feed was bad choose another
        if got is None:
            r.close()
            print("continuing on in get98")
            continue
        the98[rurl] = got
        counter += 1
        print(counter)
        r.close()
    return the98


def getData():
    data = {}
    # have a useragent so we do not look like a robot
    useragent = 'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:45.0)
    Gecko/20100101 Firefox/45.0'
    session = requests.Session()  # type: requests.Session
    session.headers.update({'User-Agent': useragent})
    try:
        # get the data
        wsdldata = consume_all(wsdl, session)
        data[wsdldata['title']] = wsdldata['text']
        fmeasuredata = consume_all(fmeasure, session)
        data[fmeasuredata['title']] = fmeasuredata['text']
        data98 = get98(session)
        # write out the uris used for the 98 other blogs
        with open("98bloguris.dat", "w+") as out:
            for url, d in data98.items():
                data[d['title']] = d['text']
                out.write("%s\n" % url)
        session.close()
        # write out our data to json
        with open("blogdata.json", "w+") as out:
            out.write(jjson.dumps(data, indent=1))

    except Exception as e:
        print(e)
        session.close()


if __name__ == "__main__":
    if not os.path.exists(os.getcwd()+"/datafiles"):
        os.makedirs(os.getcwd()+"/datafiles")
    getData()
```

Listing 2: Get Blog Feeds

Question 2

```
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

The rest of the code in processData.py seen in listing 3 deals with generating the answers to questions 2 through 4. The methods used to generate those answers are *do_non_stem()* and *do_stemmed()*. Both operate as such:

1. Generate the respective blog term matrix file
2. Read the file in using the clusters.py file(from the code accompanying the programming cognitive intelligence book.
3. Do hierarchical clustering
4. Generate text denogram and jpg denogram
5. Perform kmeans clustering and log iterations and centroid values
6. Perform dimension reduction
7. Generate dimension reduced cluster jpg

Please note that unlike the code from the book
I included stemming as with out it we are choosing the top 500 terms out of unique 2692 terms in total.

The dendrogram generated for the nonstemmed version is seen in figure 1 and the text version can be found in the datafiles folder file name, blogtop500_asciideno.txt. The dendrogram is quite large so fitting it into this report was difficult and the file name is blogtop500_deno.jpg and is found in the datafiles directory.

Of note the Web Science and Digital Libraries Research Group got grouped under Boggle Me Thursdays, which is under Floorshine Zipper Boots which happens to be under F-Measure. I find it interesting that the personal blog for the head of the WSDL is above the group its self.

The stemmed version is seen in figure 2. F-Measure is grouped with talk radio blogs and media blogs whereas the WSDL blog is grouped with blogs such as What am I doing(above) and The Night Mail(below). These results are more surprising to me especially for the WSDL blog.

```
from collections import defaultdict
from operator import add

from functional import seq
from nltk.stem.snowball import EnglishStemmer

import clusters


# this class represents the word data for a particular feed
class feed:
    # pass flag if we are to do the stemming
    def __init__(self, fentry, doStem=False):
        self.title = fentry[0]
        self.wordCount = fentry[1]
        self.stemCount = defaultdict(int)
        if doStem:
            self._stem_count()

    def _stem_count(self):
        eng = EnglishStemmer()
        for word in self.wordCount.keys():
            self.stemCount[eng.stem(word)] += 1

    def words(self):
        return set(self.wordCount.keys())

    def __str__(self):
        return "%s: %s" % (self.title, ' '.join(list(self.wordCount
    .keys())))


# fake tfidf
def filter_fun(wc):
    frac = float(wc[1]) / float(100)
    return 0.1 < frac < 0.5


# output for the non-stemmed data file
def output(f, top):
    out = [f.title]
    for wd in top:
        out.append("%d" % f.wordCount.get(wd, 0))
    return '\t'.join(out) + "\n"


# output for the stemmed data file
def output_stem(f, top):
    out = [f.title]
    for wd in top:
        out.append("%d" % f.stemCount.get(wd, 0))
    return '\t'.join(out) + "\n"


def generate_blogfile():
    # read the data in as json and then transform to feeds
    feedData = seq.json("datafiles/blogdata.json").map(lambda fe:
```

```
    feed(fe)).to_list()  # type: list[feed]
    '''
    get the top 500 words by word count over all words:
        for feed <- feeds, (word,count) <- feed.wordCount.items():
    yeild (word,count)
        keep all wordCounts > 10
        groupby + reduce word:(word1,c1),(word1,c2) -> (word1,c1,c2
    ,c3,c4) -> (word1,sumC)
        keep all wordCounts that meet fake tfidf
        transform (word,sumC) -> word
        take top 500
        transform to list
    '''
    top500 = seq(feedData).flat_map(lambda f: list(f.wordCount.
    items())) \
        .filter(lambda wc: wc[1] > 10) \
        .map(lambda wc: (wc[0], 1)) \
        .reduce_by_key(add) \
        .filter(filter_fun) \
        .order_by(lambda wc: -wc[1]) \
        .take(500) \
        .map(lambda wc: wc[0]) \
        .to_list()
    # sort alphabetically
    top500 = sorted(top500)
    print(len(top500))
    # write resultant to file
    with open("datafiles/blogtop500.txt", "w+") as out:
        out.write("Blog\t%s\n" % '\t'.join(top500))
        for tf in sorted(feedData, key=lambda f: f.title):
            out.write(output(tf, top500))


def generate_blogfile_stem():
    # same as non-stem except use stemmed data
    feedData = seq.json("datafiles/blogdata.json").map(lambda fe:
    feed(fe, True)).to_list()  # type: list[feed]
    top500 = seq(feedData).flat_map(lambda f: list(f.stemCount.
    items())) \
        .filter(lambda wc: wc[1] > 1) \
        .map(lambda wc: (wc[0],1))\
        .reduce_by_key(add) \
        .filter(filter_fun) \
        .order_by(lambda wc: -wc[1]) \
        .take(500) \
        .map(lambda wc: wc[0]) \
        .to_list()
    top500 = sorted(top500)
    print(len(top500))
    with open("datafiles/blogtop500_stemmed.txt", "w+") as out:
        out.write("Blog\t%s\n" % '\t'.join(top500))
        for tf in sorted(feedData, key=lambda f: f.title):
            out.write(output_stem(tf, top500))


def do_non_stem():
    # generate the blog file
```

```
108     generate_blogfile()
109     # read the data in
110     blognames, words, data = clusters.readfile('datafiles/
        blogtop500.txt')
111     # do clustering
112     clust = clusters.hcluster(data)
113     # write out asci denogram
114     with open("datafiles/blogtop500_asciideno.txt", "w+") as out:
115         clusters.printclust2file(clust, out, labels=blognames)
116     # generate jpg version of same denogram
117     clusters.drawdendrogram(clust, blognames, jpeg='datafiles/
        blogtop500_deno.jpg')
118     # do kmeans and log to file
119     with open("datafiles/kmeans_blogtop500.txt", "w+") as kout:
120         for k in [5, 10, 20]:
121             print("For k=%d" % k)
122             kout.write("K=%d\n" % k)
123             kout.write("Iterations\n")
124             # kmeans for value k
125             centriods = clusters.kcluster_toFile(data, k=k, out=
        kout)
126             kout.write("Centroid Values\n-------------------------\
        n")
127             # log centroid values
128             for count, centriod in enumerate(centriods, 1):
129                 print("Centroid #%d" % count)
130                 kout.write("Centroid #%d\n" % count)
131                 values = []
132                 for idx in centriod:
133                     print(blognames[idx])
134                     values.append(blognames[idx])
135                 kout.write("%s\n" % ', '.join(values))
136             kout.write("=========================================\n")
137             print("--------")
138     # do the dimensionality reduction
139     with open("datafiles/dimensionReductionNonStemmed.txt","w+") as
         dout:
140         scaled = clusters.scaledown_logiter(data,out=dout)
141     # generated the similar blog jpg
142     clusters.draw2d(scaled, blognames, jpg='datafiles/
        blogtop500_clust2d.jpg')
143
144
145 def do_stemmed():
146     generate_blogfile_stem()
147     blognames, words, data = clusters.readfile('datafiles/
        blogtop500_stemmed.txt')
148     clust = clusters.hcluster(data)
149     with open("datafiles/blogtop500stemmed_asciideno.txt", "w+") as
         out:
150         clusters.printclust2file(clust, out, labels=blognames)
151     clusters.drawdendrogram(clust, blognames, jpeg='datafiles/
        blogtop500stemmed_deno.jpg')
152
153     with open("datafiles/kmeans_blogtop500stemmed.txt", "w+") as
        kout:
154         for k in [5, 10, 20]:
```

```
155             print("For k=%d" % k)
156             kout.write("K=%d\n" % k)
157             kout.write("Iterations\n")
158             centriods = clusters.kcluster_toFile(data, k=k, out=
    kout)
159             kout.write("Centroid Values\n-------------------------\
    n")
160             for count, centriod in enumerate(centriods, 1):
161                 print("Centroid #%d" % count)
162                 kout.write("Centroid #%d\n" % count)
163                 values = []
164                 for idx in centriod:
165                     print(blognames[idx])
166                     values.append(blognames[idx])
167                 kout.write("%s\n" % ', '.join(values))
168             kout.write("=========================================\n")
169             print("--------")
170     with open("datafiles/dimensionReductionStemmed.txt", "w+") as
    dout:
171         scaled = clusters.scaledown_logiter(data, out=dout)
172     clusters.draw2d(scaled, blognames, jpg='datafiles/
    blogtop500stemmed_clust2d.jpg')
173
174
175 if __name__ == "__main__":
176     do_non_stem()
177     do_stemmed()
```

Listing 3: Process Feed Data and Generate Output

Question 3

`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 interations were required for each value of k?`

Answer

The full output of kmeans can be seen in listing 4 for non-stemmed and 5 for stemmed. For non-stemmed k=5 took 8 iterations, k=10 took 4, and k=20 took 4 as well. The stemmed version for k=5 took 6 iterations, k=10 took 5 and k=20 took 3.

```
K=5
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8
Centroid Values
-------------------------
Centroid #1
"DANCING IN CIRCLES", Boggle Me Thursday, Floorshime Zipper Boots, Interstellar Radio Shower, Parish Radio, Samtastic! Review, THE HUB, Web Science and Digital Libraries Research Group, theindiefriend,
Centroid #2
A Wife's Tale, A2 MEDIA COURSEWORK JOINT BLOG, Becky Sharp Fashion Blog, But She's Not Stupid, Deadbeat, Diagnosis: No Radio, Flatbasset, How to be an artist and still pass for normal, If You Give a Girl a Camera..., KiDCHAIR, La Fotografa Efectista Abstracta. Fotos Abstractas. Abstract Photos., Mile In Mine, My Name Is Blue Canary, One Stunning Single Egg, Our Podcast Could Be Your Life, Pithy Title Here, Rants from the Pants, Rod Shone, Room 19's Blog 2016, Sonology, Steel City Rust, Stonehill Sketchbook, T H E V O I D S, The Girl at the Rock Show, The Moon Topples, The Stearns Family, Tremagazine, Tremble Under Boom Lights, What Am I Doing?, from a voice plantation, funky little demons, isyeli's, jaaackie., sweeping the kitchen, the fast break of champions
Centroid #3
Cherry Area, Dust and Water Studios, Karl Drinkwater, Morgan's Blog, Riley Haas' blog, The Bunker, The Listening Ear, The Nosebleed Section, bittersweet, forget about it
Centroid #4
*Sixeyes: by Alan Williamson, ., 60@60 Sounding Booth, Blog Name Pending, Chantelle Swain A2 Media Studies, DaveCromwell Writes, Did Not Chart, Eli Jace | The Mind Is A Terrible Thing To Paste, Encore, Everyday Music, F-Measure, FOLK IS NOT HAPPY, I Hate The 90s, I/LOVE/TOTAL/DESTRUCTION, MAGGOT CAVIAR, MTJR RANTS & RAVES ON MUSIC, Rosie Gigg A2 Media Studies, Swinging Singles Club, The Jeopardy of Contentment, The Kids Are Coming Up From Behind, The Night Mail, The Power of Independent Trucking, The World's First Internet Baby, mouxlaloulouda, turnitup!
Centroid #5
A H T A P O T, Desolation Row Records, FlowRadio Playlists (and Blog), Green Eggs and Ham Mondays 8-10am, Haris Sfakianakis Photography, INDIEohren.!, IoTube :), KISTE F.M., Lost in the Shuffle, MARISOL, MR. BEAUTIFUL TRASH ART, MarkEOrtega's Journalism Portfolio, SEM REGRAS, Spinitron Blog, Stories From the City, Stories From the Sea, The Campus Buzz on WSOU, adrianoblog, k nstler treu, this time tomorrow,

=========================================
```

```
K=10
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Centroid Values
-------------------------
Centroid #1
Riley Haas' blog
Centroid #2
A Wife's Tale, But She's Not Stupid, Cherry Area, Deadbeat,
    Diagnosis: No Radio, Karl Drinkwater, MarkEOrtega's Journalism
    Portfolio, Mile In Mine, Morgan's Blog, One Stunning Single Egg
    , Pithy Title Here, Rants from the Pants, Sonology, Spinitron
    Blog, Steel City Rust, Stonehill Sketchbook, The Moon Topples,
    The Nosebleed Section, The Stearns Family, What Am I Doing?,
    bittersweet, forget about it, funky little demons, isyeli's,
    jaaackie., the fast break of champions
Centroid #3
KiDCHAIR, My Name Is Blue Canary, The Girl at the Rock Show,
    Tremagazine, Tremble Under Boom Lights, from a voice plantation
Centroid #4
"DANCING IN CIRCLES", Boggle Me Thursday, Interstellar Radio Shower
    , Samtastic! Review, THE HUB, sweeping the kitchen,
    theindiefriend,
Centroid #5
*Sixeyes: by Alan Williamson, ., Blog Name Pending, DaveCromwell
    Writes, Did Not Chart, Eli Jace | The Mind Is A Terrible Thing
    To Paste, Encore, Everyday Music, F-Measure, FOLK IS NOT HAPPY,
     I Hate The 90s, I/LOVE/TOTAL/DESTRUCTION, MAGGOT CAVIAR, MTJR
    RANTS & RAVES ON MUSIC, Parish Radio, Swinging Singles Club,
    The Jeopardy of Contentment, The Kids Are Coming Up From Behind
    , The Power of Independent Trucking, The World's First Internet
     Baby, mouxlaloulouda, turnitup!
Centroid #6
A2 MEDIA COURSEWORK JOINT BLOG, Chantelle Swain A2 Media Studies,
    Floorshime Zipper Boots, Rosie Gigg A2 Media Studies, T H E V O
     I D S, Web Science and Digital Libraries Research Group
Centroid #7
Becky Sharp Fashion Blog, Dust and Water Studios, The Bunker, The
    Listening Ear, The Night Mail
Centroid #8
60@60 Sounding Booth, Desolation Row Records, Flatbasset,
    INDIEohren.!, La Fotografa Efectista Abstracta. Fotos
    Abstractas. Abstract  Photos., Our Podcast Could Be Your Life
Centroid #9
A H T A P O T, FlowRadio Playlists (and Blog), Green Eggs and Ham
    Mondays 8-10am, Haris Sfakianakis Photography, IoTube      :),
    KISTE F.M., Lost in the Shuffle, MARISOL, MR. BEAUTIFUL TRASH
    ART, SEM REGRAS, Stories From the City, Stories From the Sea,
    The Campus Buzz on WSOU, adrianoblog, k nstler treu, this time
     tomorrow,
Centroid #10
How to be an artist and still pass for normal, If You Give a Girl a
     Camera..., Rod Shone, Room 19's Blog 2016
```

```
================================
K=20
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Centroid Values
--------------------------------
Centroid #1
*Sixeyes: by Alan Williamson, ., Blog Name Pending, DaveCromwell
    Writes, Did Not Chart, Eli Jace | The Mind Is A Terrible Thing
    To Paste, Encore, F-Measure, FOLK IS NOT HAPPY, I Hate The 90s,
     I/LOVE/TOTAL/DESTRUCTION, MAGGOT CAVIAR, MTJR RANTS & RAVES ON
     MUSIC, Parish Radio, Swinging Singles Club, The Jeopardy of
    Contentment, The Kids Are Coming Up From Behind, The Power of
    Independent Trucking, The World's First Internet Baby,
    mouxlaloulouda
Centroid #2
60@60 Sounding Booth, Desolation Row Records, FlowRadio Playlists (
    and Blog), Green Eggs and Ham Mondays 8-10am, INDIEohren.!,
    IoTube      :), Lost in the Shuffle, One Stunning Single Egg,
    Our Podcast Could Be Your Life, Stories From the City, Stories
    From the Sea, The Campus Buzz on WSOU, k nstler treu, this
    time tomorrow
Centroid #3

Centroid #4
La Fotografa Efectista Abstracta. Fotos Abstractas. Abstract
    Photos.
Centroid #5
Dust and Water Studios, Riley Haas' blog, The Nosebleed Section
Centroid #6
The Night Mail, turnitup!
Centroid #7
Becky Sharp Fashion Blog, Everyday Music, from a voice plantation
Centroid #8
"DANCING IN CIRCLES", THE HUB
Centroid #9
A2 MEDIA COURSEWORK JOINT BLOG, Rosie Gigg A2 Media Studies, T H E
    V O I D S
Centroid #10
Chantelle Swain A2 Media Studies, sweeping the kitchen,

Centroid #11
A H T A P O T, KISTE F.M., MARISOL, MR. BEAUTIFUL TRASH ART
Centroid #12
Haris Sfakianakis Photography,

Centroid #13
Cherry Area, Deadbeat, Diagnosis: No Radio, Flatbasset, KiDCHAIR,
    My Name Is Blue Canary, Pithy Title Here, Sonology, Steel City
    Rust, The Girl at the Rock Show, The Stearns Family, Tremble
    Under Boom Lights, bittersweet, funky little demons, isyeli's,
    the fast break of champions
Centroid #14
```

```
Floorshime Zipper Boots, Tremagazine
Centroid #15
Boggle Me Thursday, Interstellar Radio Shower, Samtastic! Review, Web Science and Digital Libraries Research Group, theindiefriend
Centroid #16
MarkEOrtega's Journalism Portfolio, Spinitron Blog
Centroid #17
SEM REGRAS, adrianoblog
Centroid #18

Centroid #19
A Wife's Tale, But She's Not Stupid, How to be an artist and still pass for normal, Karl Drinkwater, Mile In Mine, Morgan's Blog, Rants from the Pants, Room 19's Blog 2016, Stonehill Sketchbook, The Bunker, The Listening Ear, The Moon Topples, What Am I Doing?, forget about it, jaaackie.
Centroid #20
If You Give a Girl a Camera..., Rod Shone
==========
```

Listing 4: Kmeans output non-stemmed

```
K=5
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Centroid Values
----------
Centroid #1
., DaveCromwell Writes, Encore, FOLK IS NOT HAPPY, Floorshime Zipper Boots, I Hate The 90s, I/LOVE/TOTAL/DESTRUCTION, MTJR RANTS & RAVES ON MUSIC, Mile In Mine, Rosie Gigg A2 Media Studies, Samtastic! Review, Sonology, The Kids Are Coming Up From Behind, The Night Mail, from a voice plantation, mouxlaloulouda
Centroid #2
A Wife's Tale, Boggle Me Thursday, Dust and Water Studios, Eli Jace | The Mind Is A Terrible Thing To Paste, Everyday Music, How to be an artist and still pass for normal, MAGGOT CAVIAR, Morgan's Blog, Our Podcast Could Be Your Life, Rants from the Pants, Rod Shone, Room 19's Blog 2016, Swinging Singles Club, The Bunker, The Stearns Family, isyeli's, jaaackie.
Centroid #3
A2 MEDIA COURSEWORK JOINT BLOG, Becky Sharp Fashion Blog, But She's Not Stupid, Chantelle Swain A2 Media Studies, Cherry Area, Deadbeat, Diagnosis: No Radio, F–Measure, Flatbasset, Karl Drinkwater, MarkEOrtega's Journalism Portfolio, My Name Is Blue Canary, Pithy Title Here, Riley Haas' blog, Spinitron Blog, Steel City Rust, T H E V O I D S, The Listening Ear, The Moon Topples, The Nosebleed Section, Tremble Under Boom Lights, Web Science and Digital Libraries Research Group, What Am I Doing?,
```

```
    bittersweet
Centroid #4
"DANCING IN CIRCLES", *Sixeyes: by Alan Williamson, 60@60 Sounding
    Booth, A H T A P O T, Desolation Row Records, FlowRadio
    Playlists (and Blog), Green Eggs and Ham Mondays 8-10am, Haris
    Sfakianakis Photography, INDIEohren.!, If You Give a Girl a
    Camera..., IoTube :), KISTE F.M., La Fotografa Efectista
    Abstracta. Fotos Abstractas. Abstract Photos., Lost in the
    Shuffle, MARISOL, MR. BEAUTIFUL TRASH ART, One Stunning Single
    Egg, Parish Radio, SEM REGRAS, Stonehill Sketchbook, Stories
    From the City, Stories From the Sea, THE HUB, The Campus Buzz
    on WSOU, adrianoblog, forget about it, funky little demons,
    k nstler treu, sweeping the kitchen, theindiefriend, this time
     tomorrow, turnitup!,
              ,
Centroid #5
Blog Name Pending, Did Not Chart, Interstellar Radio Shower,
    KiDCHAIR, The Girl at the Rock Show, The Jeopardy of
    Contentment, The Power of Independent Trucking, The World's
    First Internet Baby, Tremagazine, the fast break of champions
==========
K=10
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Centroid Values
----------
Centroid #1
A Wife's Tale, Becky Sharp Fashion Blog, MTJR RANTS & RAVES ON
    MUSIC, The Jeopardy of Contentment, The Moon Topples, The Power
     of Independent Trucking, Tremble Under Boom Lights, Web
    Science and Digital Libraries Research Group, from a voice
    plantation
Centroid #2
Dust and Water Studios, KiDCHAIR, La Fotografa Efectista
    Abstracta. Fotos Abstractas. Abstract Photos., Parish Radio,
    Rod Shone, THE HUB, The Kids Are Coming Up From Behind, forget
    about it, turnitup!
Centroid #3
., Cherry Area, Flatbasset, Pithy Title Here, Riley Haas' blog, The
     Listening Ear, The Night Mail, The Nosebleed Section, The
    World's First Internet Baby, What Am I Doing?, mouxlaloulouda
Centroid #4
A2 MEDIA COURSEWORK JOINT BLOG, Deadbeat, F-Measure, MAGGOT CAVIAR,
     Mile In Mine, Sonology
Centroid #5
Chantelle Swain A2 Media Studies, Did Not Chart, Encore, I/LOVE/
    TOTAL/DESTRUCTION, MarkEOrtega's Journalism Portfolio, Rosie
    Gigg A2 Media Studies, T H E V O I D S
Centroid #6
But She's Not Stupid, DaveCromwell Writes, Diagnosis: No Radio, I
    Hate The 90s, Karl Drinkwater, My Name Is Blue Canary,
    bittersweet
```

```
Centroid #7
60@60 Sounding Booth, FOLK IS NOT HAPPY, Floorshime Zipper Boots,
    Interstellar Radio Shower, Samtastic! Review, Steel City Rust,
    The Campus Buzz on WSOU, Tremagazine, funky little demons
Centroid #8
Blog Name Pending, Morgan's Blog, Our Podcast Could Be Your Life,
    Spinitron Blog
Centroid #9
"DANCING IN CIRCLES", A H T A P O T, Desolation Row Records,
    Everyday Music, FlowRadio Playlists (and Blog), Green Eggs and
    Ham Mondays 8-10am, Haris Sfakianakis Photography, INDIEohren
    .!, If You Give a Girl a Camera..., IoTube      :), KISTE F.M.,
    Lost in the Shuffle, MARISOL, MR. BEAUTIFUL TRASH ART, SEM
    REGRAS, Stonehill Sketchbook, Stories From the City, Stories
    From the Sea, adrianoblog, k nstler treu, theindiefriend,
                                                        ,

Centroid #10
*Sixeyes: by Alan Williamson, Boggle Me Thursday, Eli Jace | The
    Mind Is A Terrible Thing To Paste, How to be an artist and
    still pass for normal, One Stunning Single Egg, Rants from the
    Pants, Room 19's Blog 2016, Swinging Singles Club, The Bunker,
    The Girl at the Rock Show, The Stearns Family, isyeli's,
    jaaackie., sweeping the kitchen, the fast break of champions,
    this time tomorrow
==========
K=20
Iterations
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Centroid Values
----------
Centroid #1
Flatbasset, I Hate The 90s, My Name Is Blue Canary, Samtastic!
    Review, The Jeopardy of Contentment, Tremagazine
Centroid #2
*Sixeyes: by Alan Williamson, Did Not Chart, Everyday Music, MAGGOT
     CAVIAR, The Campus Buzz on WSOU, funky little demons, turnitup
    !
Centroid #3
How to be an artist and still pass for normal, Rod Shone, Sonology,
     Stonehill Sketchbook, THE HUB
Centroid #4
F-Measure, Rants from the Pants
Centroid #5
I/LOVE/TOTAL/DESTRUCTION, MTJR RANTS & RAVES ON MUSIC, Swinging
    Singles Club, bittersweet, forget about it
Centroid #6
Pithy Title Here, The Moon Topples, The Nosebleed Section, The
    Stearns Family, Tremble Under Boom Lights, Web Science and
    Digital Libraries Research Group, from a voice plantation
Centroid #7
Room 19's Blog 2016, The Kids Are Coming Up From Behind, What Am I
    Doing?
Centroid #8
```

```
60@60 Sounding Booth, A H T A P O T, Desolation Row Records, Haris Sfakianakis Photography, IoTube :), KISTE F.M., La Fotografa Efectista Abstracta. Fotos Abstractas. Abstract Photos., Lost in the Shuffle, adrianoblog, theindiefriend, this time tomorrow,
Centroid #9
FlowRadio Playlists (and Blog), Green Eggs and Ham Mondays 8-10am, MR. BEAUTIFUL TRASH ART, SEM REGRAS, Steel City Rust
Centroid #10
Becky Sharp Fashion Blog, But She's Not Stupid, The Bunker, The Girl at the Rock Show, The World's First Internet Baby
Centroid #11
If You Give a Girl a Camera..., Morgan's Blog, Stories From the City, Stories From the Sea, isyeli's, the fast break of champions
Centroid #12
A Wife's Tale, Blog Name Pending, Eli Jace | The Mind Is A Terrible Thing To Paste, Interstellar Radio Shower, KiDCHAIR, Our Podcast Could Be Your Life, jaaackie., mouxlaloulouda
Centroid #13
Deadbeat, Mile In Mine, One Stunning Single Egg, sweeping the kitchen
Centroid #14
., Cherry Area, Diagnosis: No Radio, The Listening Ear, The Night Mail
Centroid #15
Parish Radio,
Centroid #16
A2 MEDIA COURSEWORK JOINT BLOG, Chantelle Swain A2 Media Studies, Rosie Gigg A2 Media Studies, T H E V O I D S
Centroid #17
DaveCromwell Writes, Dust and Water Studios, FOLK IS NOT HAPPY, Floorshime Zipper Boots
Centroid #18
Boggle Me Thursday
Centroid #19
"DANCING IN CIRCLES", INDIEohren.!, MARISOL, Spinitron Blog, k nstler treu
Centroid #20
Encore, Karl Drinkwater, MarkEOrtega's Journalism Portfolio, Riley Haas' blog, The Power of Independent Trucking
==========
```

Listing 5: Kmeans output stemmed

Question 4

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

Answer

The number of iterations required for dimension reduction for both the non-stemmed and stemmed versions can bee seen below in listings 6 and 7.

```
1 Dimension Reduction took 361 iterations
```

Listing 6: Dimension Reduction Iterations Non-Stemmed

```
1 Dimension Reduction took 75 iterations
```

Listing 7: Dimension Reduction Iterations Stemmed

The difference in the number of iterations for the dimension reduction between the non-stemmeed and stemmed blog term matrices in my opion is due to the sparse nature of the terms. The raw terms are very different as the blogs gotten are all over the place in terms of content thus it will take longer. Whereas the stemmed version the context of the words is removed and a better approximation can be made.

The resultant jpg pictures can be seen in figure 3 for non-stemmed and 4 for stemmed. Once again these pictures are way to large to clearly see in this report. For clearer inspect the files are found in datafiles/blogtop500_clust2d.jpg and datafiles/blogtop500stemmed_clust2d.jpg.

Of note in blogtop500_clust2d.jpg, the WSDL blog is grouped with Stories From the City, Stories from the Sea and F-Measure is grouped with The Kids are Coming Up From Behind.

Also of note in blogtop500stemmed_clust2d.jpg, the WSDL blog is grouped with The Night Mail and F-Measure is grouped with MarkEOrtega's Journalism Portfolio.

Question 5

```
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 500
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 500 terms and instead
come up with TFIDF scores for all the terms and then choose the top
500 from that list, but I'm trying to limit the amount of work
necessary.
w many iterations were required?

======================================================================
========The questions below is for 5 points extra credit==========
======================================================================
```

Answer

Not attempted.

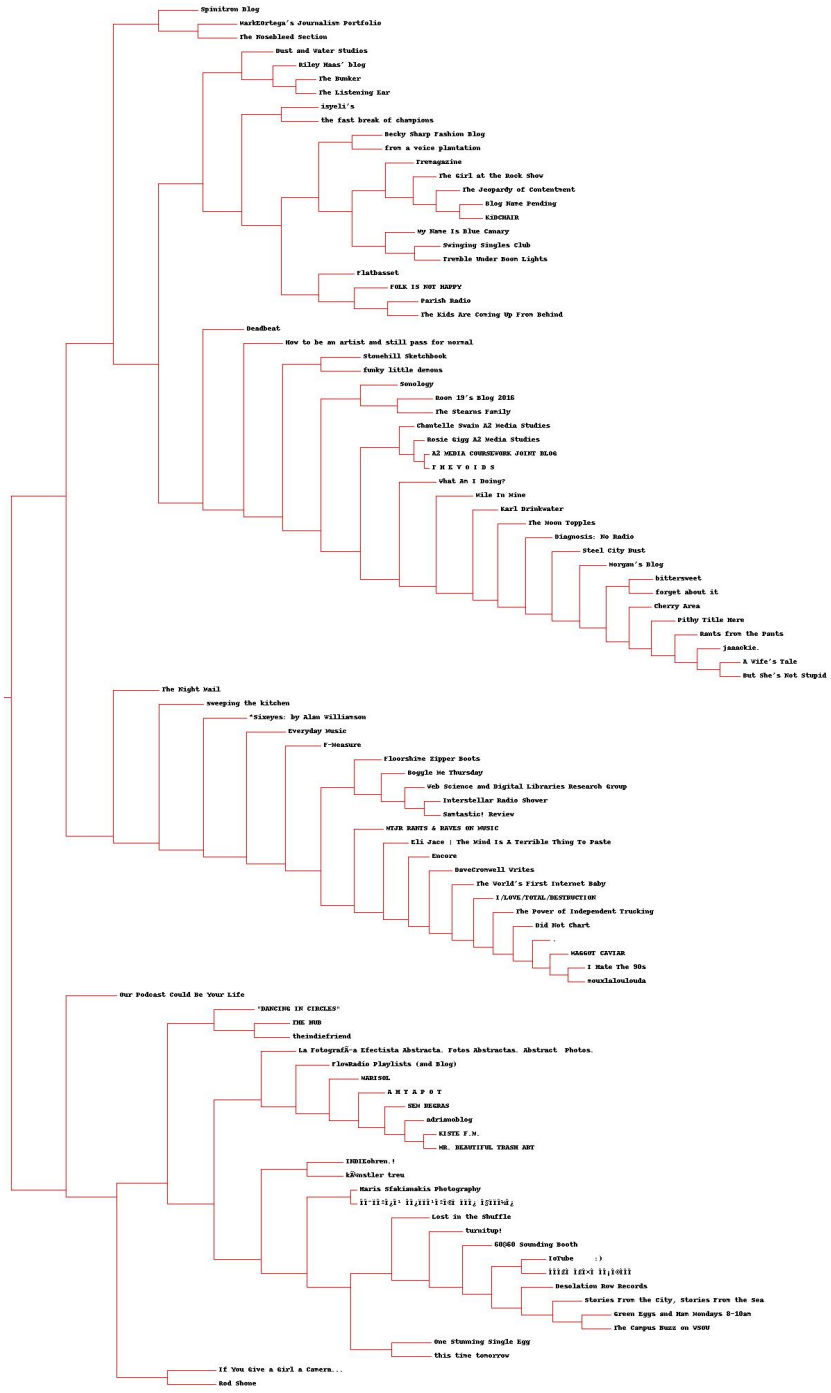

Figure 1: Dendogram of the blogs using top 500 terms

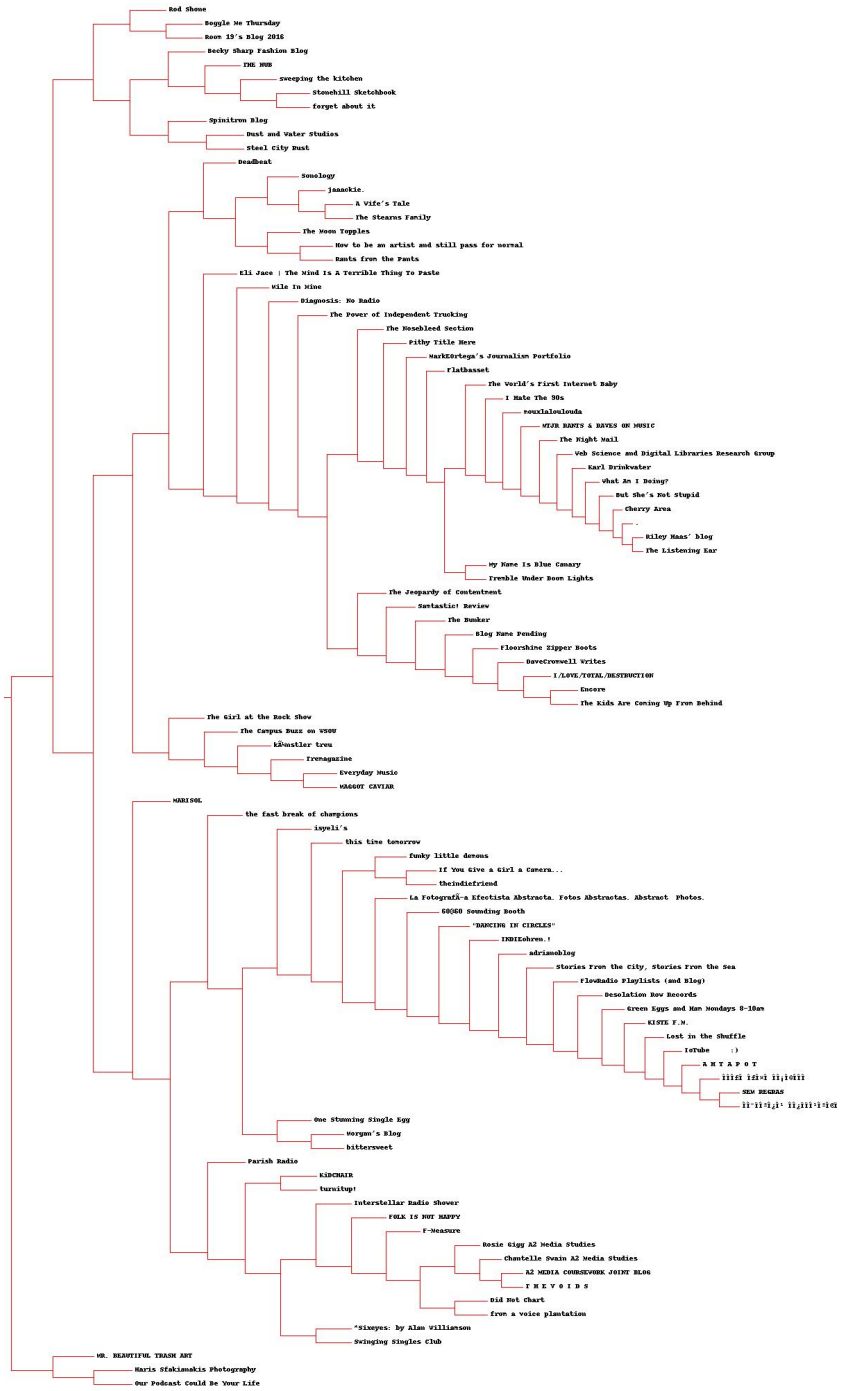

Figure 2: Dendogram of the blogs using top 500 stemmed terms

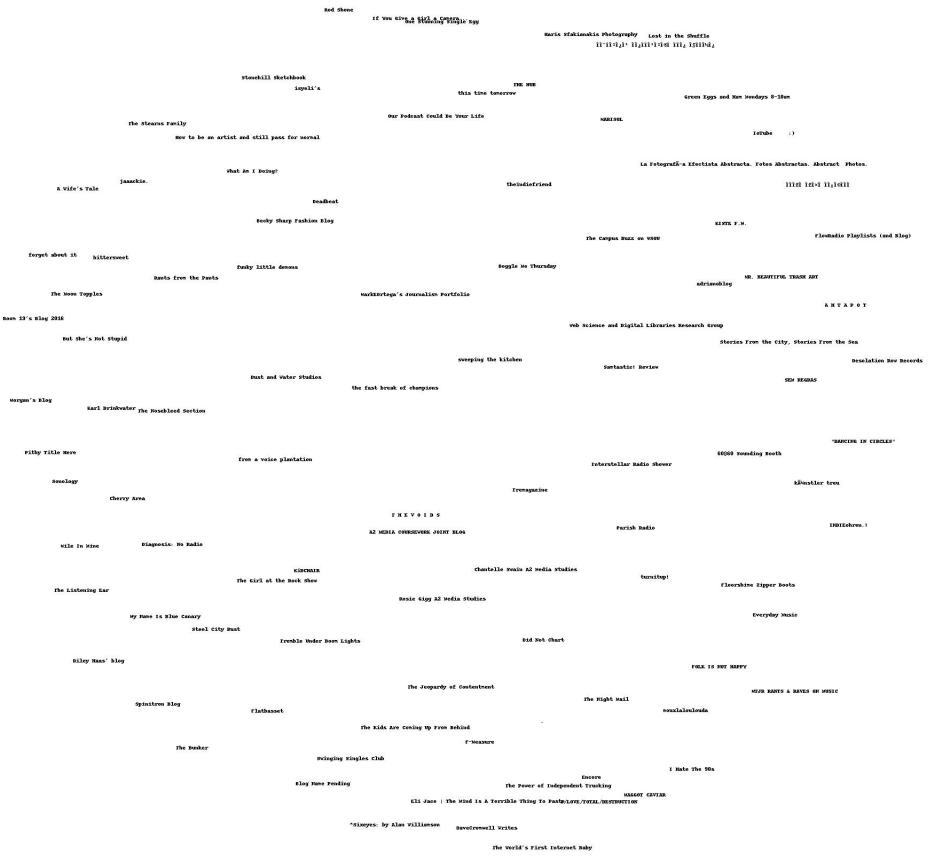

Figure 3: Dimension Reduction Non Stemmed

Figure 4: Dimension Reduction Stemmed