# Assignment 9

CS595

Introduction to Web Science

Old Dominion University

Computer Science

Due: 11:59 pm Dec 04

Lulwah Alkwai

(10 points; 2 points for each question and 2 points for aesthetics)

Support your answer: include all relevant discussion, assumptions, examples, etc.

## 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.

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 One-

To solve this question I have done the following steps:

- I grabbed 100 blogs, I started out with the two required blogs and then added some other blogs from my office friends and went from there and grabbed more using the method mentioned in class.The result was blogdata1.txt.

- For the second step which is creating a matrix I used the code generatefeedvector.py and made the required adjustments.

  The matrix required was using blog title as the identifier for each blog row and using the terms for the columns of the matrix;the values are the frequency of occurrence.

  The adjustment that were made were that the TFIDF is calculated as the score that balances the frequency of a term in a document vs. frequency of a term in all the documents.Also limit the number of terms to the most "popular" 500 terms.

  note:to run the code I had to download the feedparser using the command:

  ```
  pip install feedparser
  ```

## Question 2

2. Create an ASCII and JPEG dendrogram that clusters (i.e., HAC) the most similar blogs (see slides 12 and 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 Two-

I used the code clusters.py. However some files needed to be downloaded on my computer such as PIL. To do so I had to install clang, so I downloaded the command line tools from Xcode. Then I had to download the PIL file from `http://www.pythonware.com/products/pil/`. Finally I used the following commands:

```
brew install libjpeg
pip install PIL
```

The resulting graph is shown at figure 1. The closest blog to f-measure was the week-end.And the closest blog to the ws-dl blog is the orcale blog.

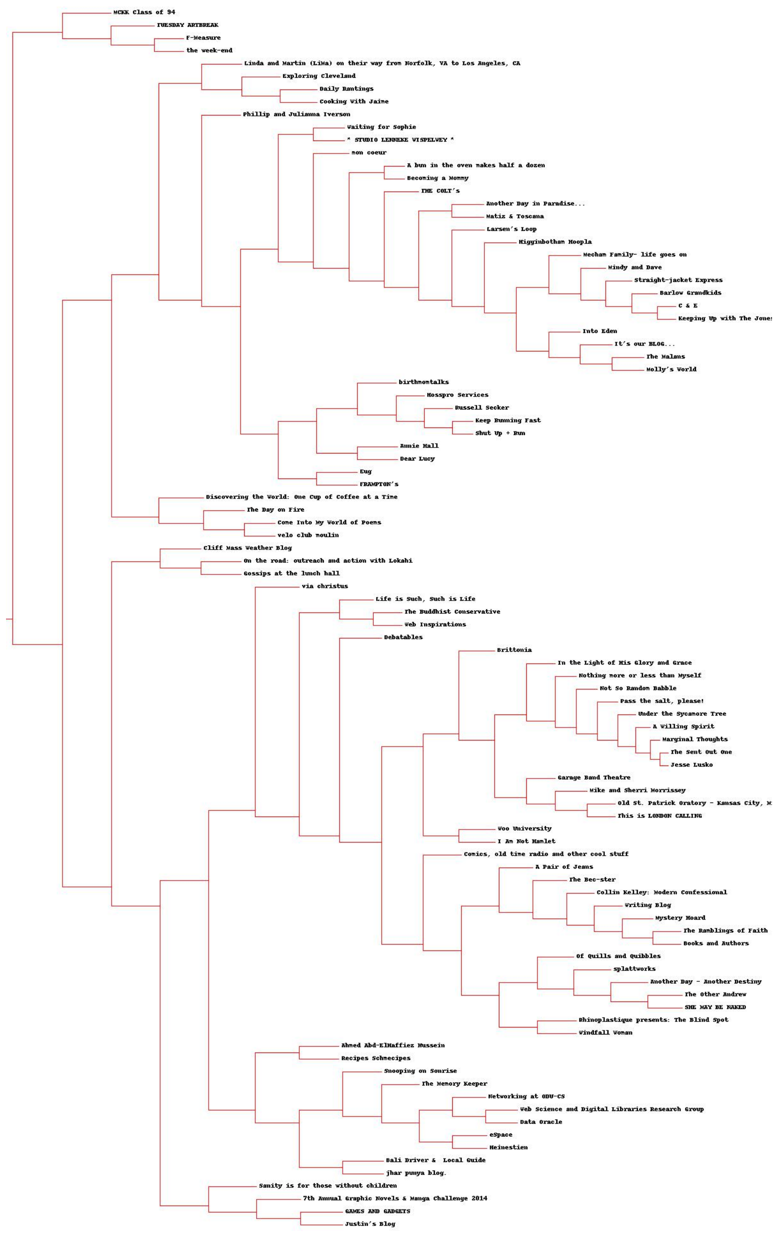

Figure 1: blog clustuer

## Question 3

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

## Answer Three-

Again using the code clusters.py I added the K-Means.The iterations were as following:

```
K=5
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6

K=10
Iteration 0
Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
Iteration 7
Iteration 8

K=20
Iteration 0
Iteration 1
Iteration 2
Iteration 3
```

So for K=5 there was 7 iterations, for K=10 there were 9 iterations and for K=20 there are 4 iterations.

## Question 4

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

## Answer Four-

Again using the code clusters.py I added the MDS to create a JPEG of the blogs.The code used was from the class slides. The number of iteration is 419. As seen in the OutputData.txt. The resulting graph is figure 2.

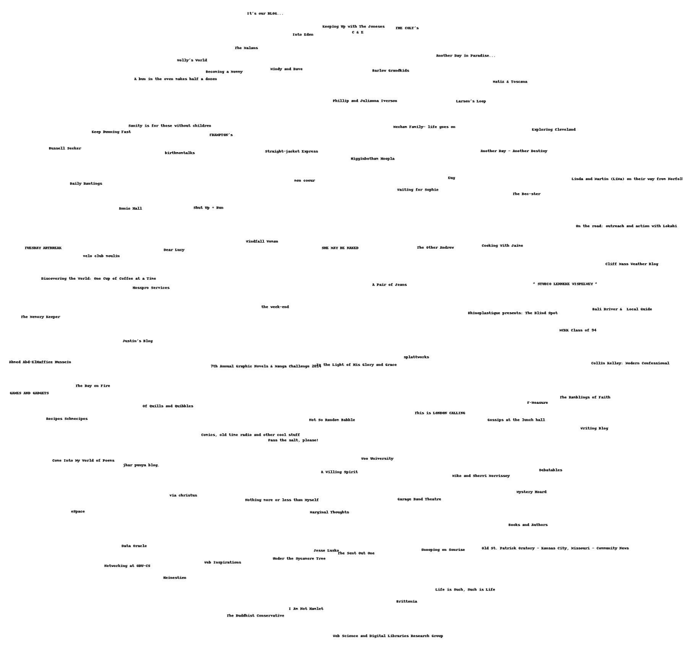

Figure 2: MDS to create a JPEG

## Question 5

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

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.

**Answer Five-**

No attempt.