36-350: Data Mining

Lab 1
Date: August 29, 2003 **Due: end of lab**

Interspersed throughout this lab are questions that you will have to answer at check-off.

1. Download the files for this lab from the course web page to the desktop:

   `http://www.stat.cmu.edu/~minka/courses/36-350/lab/`

2. Open a Word or Notepad document to record your work.

Start R

3. `Start -> All Programs -> Class software -> R 1.7.0`

4. Load the special functions for this lab:

   `File -> Source R code...`

   Browse to the desktop and pick `lab1.r` (it may have been renamed to `lab1.r.txt` when you downloaded it). Another window will immediately pop up for you to pick the `mining.zip` file you downloaded.

Processing a text file

5. Using the functions described below, read the file "politics3.txt" into R. *What is the 57th word in the document?*

The full dataset

6. The full dataset is in a table called `doc`, containing 9 posts to `talk.politics.misc` and 9 posts to `talk.religion.misc`. The `dim` function will tell you the size of the table:

   ```
   dim(doc)
   ```

   A vector `doc.labels` is also defined.

7. According to `doc`, how many times does 'the' occur in `politics3`? Give a command to select the answer from `doc`. (Note that both the rows and columns have names.)

Document similarity

8. In preparation for computing distances, remove the words which occur only once, and weight the remaining words by inverse document frequency (IDF). These two commands do it:

   ```
   doc = remove.singletons(doc)
   doc = idf.weight(doc)
   ```

   `doc` is modified in place.

9. Compute a distance matrix between all documents, using IDF and without normalizing for document length. Save the first five rows in your Word file, for use in the homework. You can also write the entire table to a file, via

   ```
   write.array(d,file="d.txt")
   ```

10. Compute a distance matrix where the weighted word counts have been normalized by the document total. Save the first five rows.

11. Compute a distance matrix where the weighted word counts have been normalized by the document's Euclidean length. Save the first five rows.

12. Pick one of the above matrices and make a multidimensional scaling visualization of it. You can save the plot by pasting into your Word document.

13. You can now get checked off.

Reading a document into R If the Desktop has a file called "politics3.txt", you can read it into R via

```
txt = read.doc("politics3.txt")
```

This makes `txt` into a vector of words. The *n*th word can obtained by typing `txt[n]`. `read.doc` removes the message header, removes all punctuation and capitalization, and converts all numbers to the pound sign `#`. `table(txt)` is the bag of words representation.

Selecting from a table or matrix When a table has two dimensions, like `doc` does, you select from it by giving a row name or number and a column name or number, e.g. `doc[2,4]` or `doc[4,"the"]`. An entire row or column can be selected by leaving out the index, e.g. `doc["politics3",]` or `doc[,4]`.

Computing distances If `x` has your (normalized and weighted) word counts, type

```
d = distances(x)
```

and `d` will become a matrix of distances.

Normalizing documents To divide each document's count vector by its sum:

```
x = div.by.sum(doc)
```

The result is put into `x` so that the original `doc` can still be used. To instead divide each document's count vector by its Euclidean length:

```
x = div.by.euc.length(doc)
```

Multidimensional scaling If `d` is a distance matrix and `doc.labels` contains the document labels (for color-coding):

```
mds(d,doc.labels)
```

This will open a window for the plot. Using the menu, select `Windows -> Tile` to arrange the sub-windows. You can also do multidimensional scaling with three dimensions instead of two, so that distances can be represented more accurately.

```
mds(d,doc.labels,k=3)
```

This gives a 3D scene that you can rotate with the mouse.