STA 242: Assignment 4

Scott Frost and Tom Shapland

March 18, 2011

In this assignment we manipulated text data using regular expressions in order to import the text data into R. Once the text data was in R, we explored features of the data using regular expressions and network plots. Our work focused on the Enron email data set.

1. Description of problems

First, we will describe the problems we encountered as we designed the code and our strategies for solving the problems.

We wanted to extract the header information (sender, the recipient, the data, and the subject) and the body of the email. While it was easy to identify the lines containing this information using regular expressions, it was difficult to extract the information. We were advised to use the read.dcf() function, which befuddled Tom. Scott figured out how to use the read.dcf() command, but Tom decided to extract the data in a different manner. We will describe how Scott used the read.dcf() command and the alternative method that Tom used.

In Scott's method, we got around this problem doing x.

In Tom's method, we used regular expressions to identify lines in the data starting with the string “From: ”, which contained the sender information. In regular emails, the sender line is preceded by the date line, and followed by the recipient line and the subject line. Once we identified the sender lines, we indentified the date, recipient, and subject lines by their position relative to the sender line. The contents of the sender, date, recipient, and subject lines were extracted. Tom’s method ignored the email body.

In some cases the emails were malformed. In other words, the recipient line did not follow the sender line. If a line that follows a sender line does not start with “To: “, then the line was identified as coming from a malformed email. Malformed emails were culled. We may have lost important information in our culled emails, but hopefully we preserved enough material to indict Chairman Ken. Using Tom’s method, the total number of emails was X and the number of malformed emails was Y. Using Scott’s method, .

It was difficult to extract the header information and email bodies without running into memory problems. There were 27,390,842 lines in the Enron email data set. Initially Scott's method required reading in the whole data set in order to extract the header information and the email body. R was unable to read in 27 million lines.

Scott got around this problem doing x.

Tom also ran into memory problems with his method. Tom resolved this problem by reading 10,000 line segments of the data set into R, extracting the header information, outputting the header information in a file, then repeating the loop.

Only a small portion of our time was spent developing usable code. For most of our work in this assignment we did not use version control because we only had a few lines of quasi-functional code. Once we had a small amount of functional code, we used version control.

2. Interesting features of the Enron dataset

We will not describe interesting features of the Enron data set. We created 2 network maps. The first network map describes the entire email data set. The second network map describes the email communications involving Ken Lay.

We also explored the email data using regular expressions. We search Ken Lay’s emails for words such as “fraud” and “greed”, and text strings such as “nuts to the employee penchants!”.

3. Interesting features of the R-mailing list

Because we encountered so much difficulty in processing the Enron email data set, we did not have time to explore the R-mailing list.