**IST 687 PREP EXERCISE 11**

**Student Name: Bhavish Kumar**

**Prep Exercise No: 11**

**Date Due: 13th November 2019**

1. **Getting Ready: Loading the data.**
   1. Install the “tm” package and library it.

**install.packages("tm")**

**library(tm)**

* 1. On Blackboard download the speech.txt, positive-words.txt, and negative-words.txt files. Set the working directory to point to the folder that contains the downloaded files.

## Downloaded the files and saved to the directory C:\\Users\\LENOVO\\Desktop\\SYR ADS\\Sem 1\\IST 687 Intro to Data Science\\Prep Exercise & Homework"

* 1. Read in the speech.txt file using the *scan()* function, the code is provided below.

**charVector <- scan("speech.txt", character(0), sep = "\n")**

* 1. Using similar code read in the positive-words.txt and negative-words.txt files as “posWords” and “negWords” respectively. Be sure to remove any header information from those files (if you need help, review chapter 15 in the text book).

**posWords <- scan("positive-words.txt", character(0), sep = "\n")   
## Saving positive words text file to vector**

**posWords <- posWords[-1:-34] ## Removing first 34 rows**

**negWords <- scan("negative-words.txt", character(0), sep = "\n")   
## Saving Negative words text file to vector**

**negWords <- negWords[-1:-34] ## Removing first 34 rows from vector**

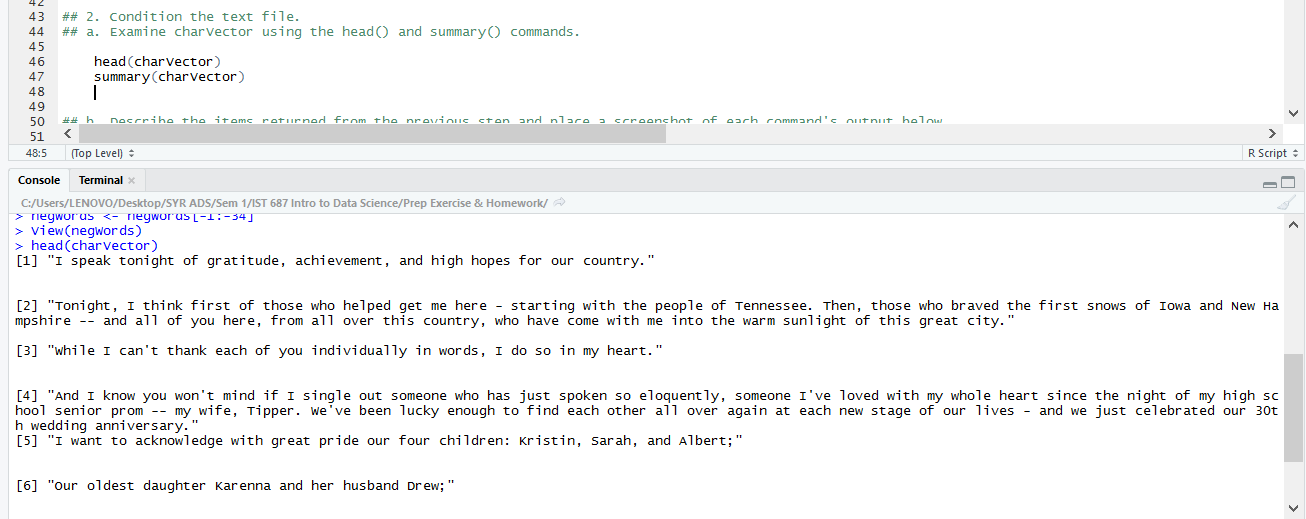
1. **Condition the text file.** 
   1. Examine charVector using the *head()* and *summary()* commands.

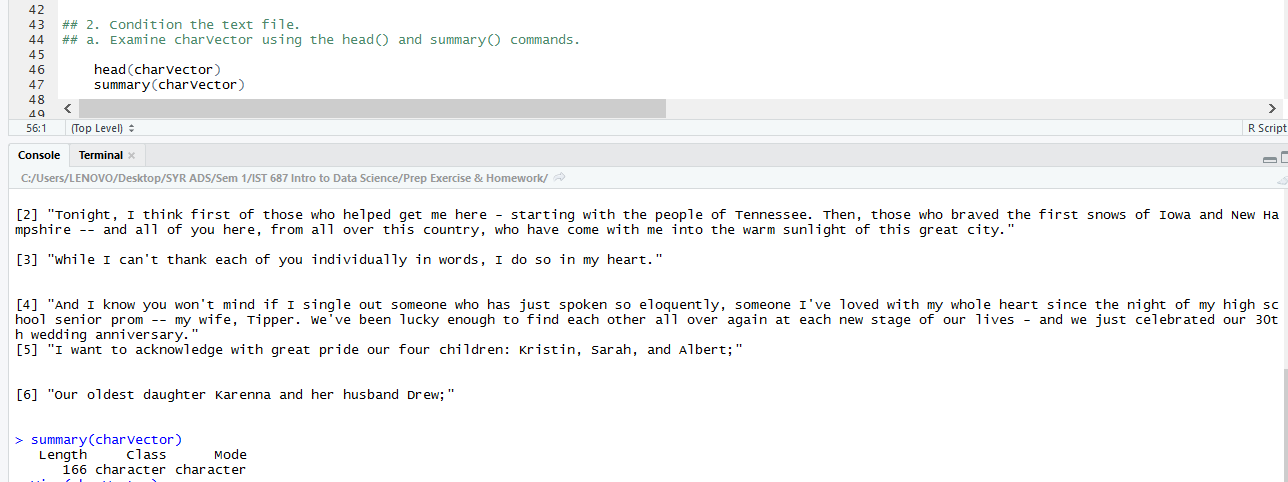
**head(charVector)**

**summary(charVector)**

* 1. Describe the items returned from the previous step and place a screenshot of each command’s output below (hint: There should be 166 total items in the charVector).

**The head function returns the first 6 rows of the vector where each row contains one whole sentence from the text file.**



**The summary function tells us the number of rows in the vector, the class and the datatype/mode.**

1. **Transform charVector into a term document matrix.**
   1. In a few sentences define/explain a term document matrix.

**A Term Document Matrix is used to convert the loaded text vector into a rectangular matrix data structure with rows and columns where the rows appear as terms and documents appear as columns. If a term "analyst" appears once in document 1 and thrice in document 2, then the columns for term "analyst" will contain 1, 3.**

* 1. Create a word corpus (called words.corpus). Then, make sure everyting is lower case, remove punctuation, remove numbers and then, finally, remove English stopwords. If you need help, review pages 180-181 in the text book (**Note:** Ignore any warning messages that come from *tm\_map()*)

**words.vec <- VectorSource(charVector)**

**words.corpus <- Corpus(words.vec)**

**words.corpus <- tm\_map(words.corpus, content\_transformer(tolower))   
## Making all the letters lower case**

**words.corpus <- tm\_map(words.corpus, removePunctuation)   
 ## Removing all the punctuations**

**words.corpus <- tm\_map(words.corpus, removeNumbers)   
## Removing all the numbers**

**words.corpus <- tm\_map(words.corpus, removeWords, stopwords("english"))   
## Removing all the stop words such as 'the', 'a', 'an', 'at' etc. which are useless for differentiating between the documents**

* 1. Create a TermDoumentMatrix variable called ‘tdm’, from the words.corpus variable.

**tdm <- TermDocumentMatrix(words.corpus)   
## Creating a Term Document Matrix from the words.corpus using the function**

1. **Understanding the term document matrix.**
   1. Using *inspect(tdm*) function create a summary of the term document matrix along with a sample of some of the terms and documents.

**inspect(tdm)   
## Using the inspect() function to view the summary of the term document matrix**

* 1. In a few sentences, explain the output from the previous step. Place a screenshot of the output from *inspect(tdm)* below.

***## In the previous step we obtained a summary of our term document matrix using the inspect function.***

***## From the summary we can observe that the function has created 1211 terms and 166 documents because of which our matrix will have 1211 rows and 166 columns.***

***## The Non Sparse entries parameter shows us that out of 198452 cells in the matrix (1211 rows and 166 columns) there are only 2574 cells with non zero entries.***

***## The weighting parameter shows that we use the term frequency in each cell, i.e. it indicates the number of occurences of a given term across each document.***

***## We also get to see a small sample of some of the terms and their documents.***

