# Prep Exercise (PE11) Sentiment Analysis

### General Instructions

1. For this exercise you will answer all of the questions in this document and turn it in to Blackboard.
2. Before you get started make sure to read Chapters 14 and 15 of *An Introduction to Data Science* and execute the code throughout the chapter to gain familiarity.
3. Getting Started:
   1. The Internet contains billions of unstructured documents that consist mainly of natural language text. For example, presidency.ucsb.edu contains hundreds of speeches made by presidents and presidential nominees. Locating and parsing text documents, restructuring them, filtering them, counting particular words or phrases, and creating visualizations – these are all critical skills for data scientists. The tm package provides an essential toolbox for manipulating text data.
   2. As usual we will use this Prep Ex to set you up for the homework exercises and test your knowledge of materials within the chapter reading. Let’s begin…

# IST 687, Standard Homework Heading

#

# Student name: Thadhani Hitesh Chandrakumar

# Homework number: PE11

# Date due: 13th Nov 2019 11:59 PM

#

# Attribution statement: (choose the statements that are true)

# 1. I did this work by myself, with help from the book and the professor

# 2. I did this work with help from the book and the professor and these Internet sources

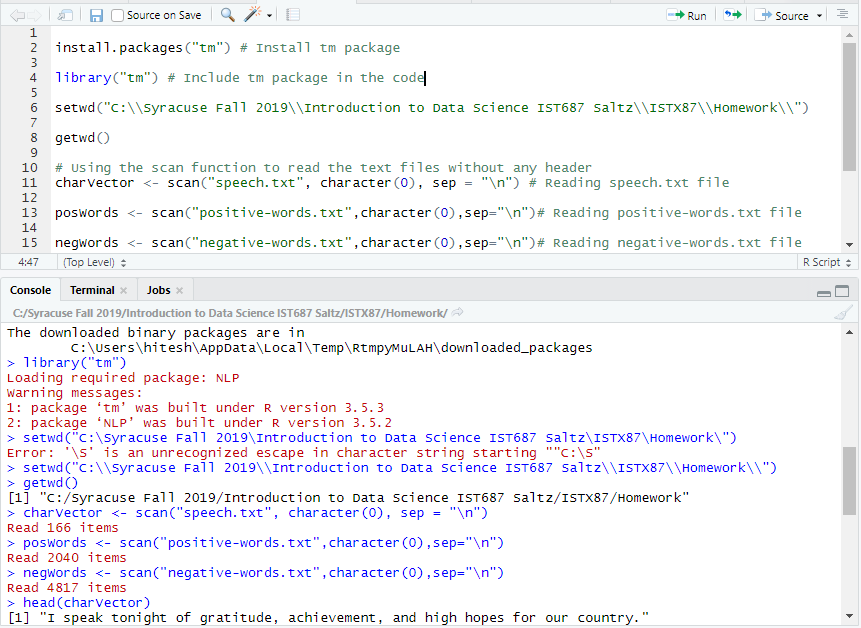
# 3. I did this work with coaching from <Name of another student> but did not cut and paste any code

### Prep Exercise

1. **Getting Ready: Loading the data.**
   1. Install the “tm” package and library it.
   2. 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.
   3. Read in the speech.txt file using the *scan()* function, the code is provided below.

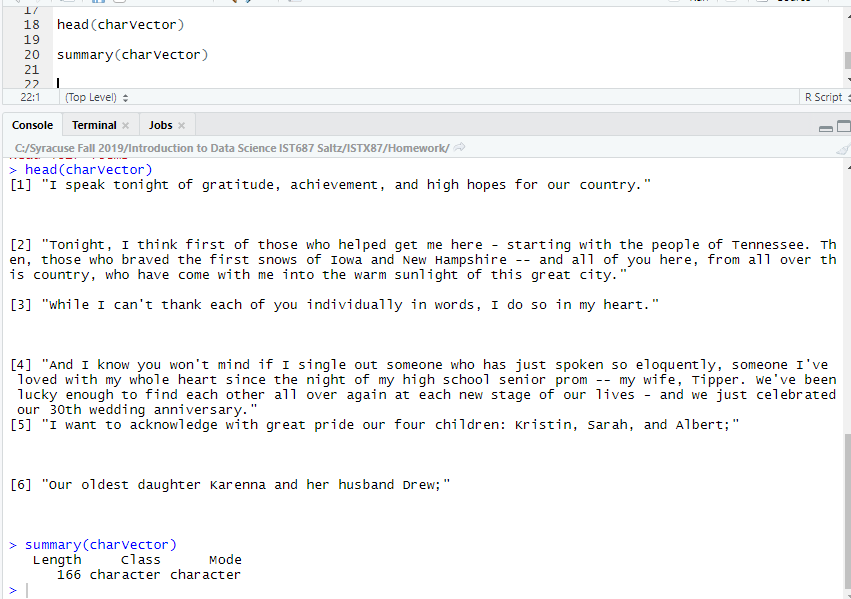
**charVector <- scan("speech.txt", character(0), sep = "\n")**

* 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).



1. **Condition the text file.** 
   1. Examine charVector using the *head()* and *summary()* commands.
   2. 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 command shows the 1st 6 lines of the text file separated by new line character while the loading the txt file. The summary command shows the length of the document like 166 here and the class and mode. It shows there are 166 terms in charVector number of rows in that document where each row of 166 is a whole text line.

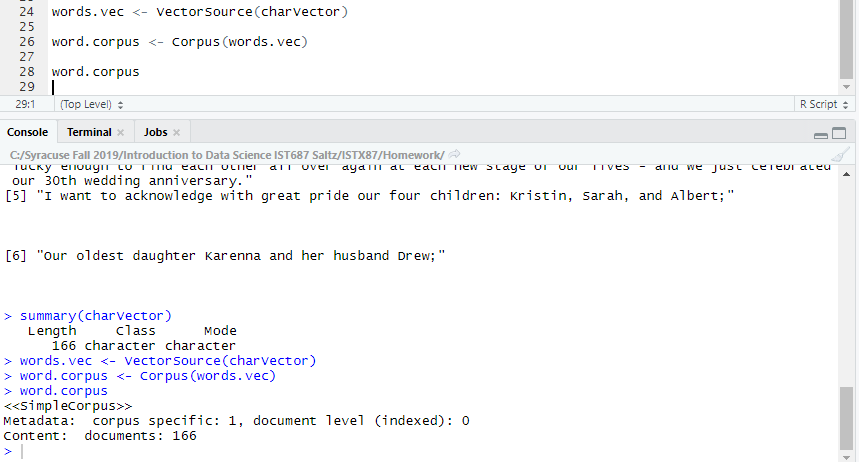


1. **Transform charVector into a term document matrix.**
   1. In a few sentences define/explain a term document matrix.

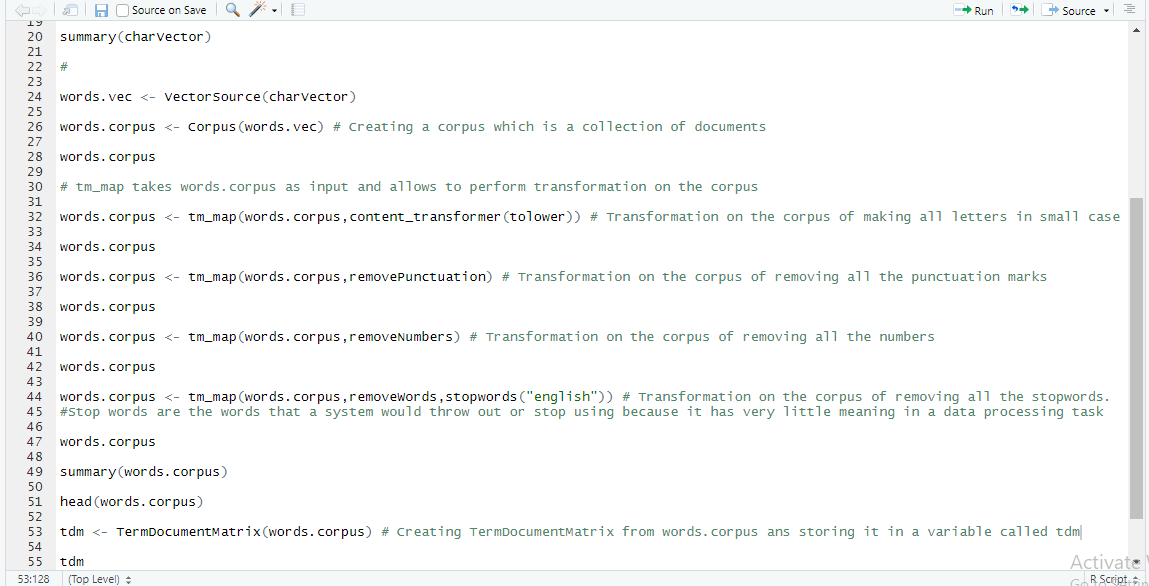
Term Document matrix is a rectangular data structure where rows are the terms and the columns are the documents. It represents the words in a text as a table (matrix) of the numbers. The matrix has 0 for the absence of the particular word in the text whereas 1 for the word is present. So the rows represent the text to be analyzed and the columns would have individual words from that text to be analyzed. Below is the example of the term document matrix.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **I** | **like** | **reading** | **Go** | **For** | **Walk** | **in** | **snow** |
| **I like reading** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** |
| **I go for walk in snow** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **1** |

* 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()*)

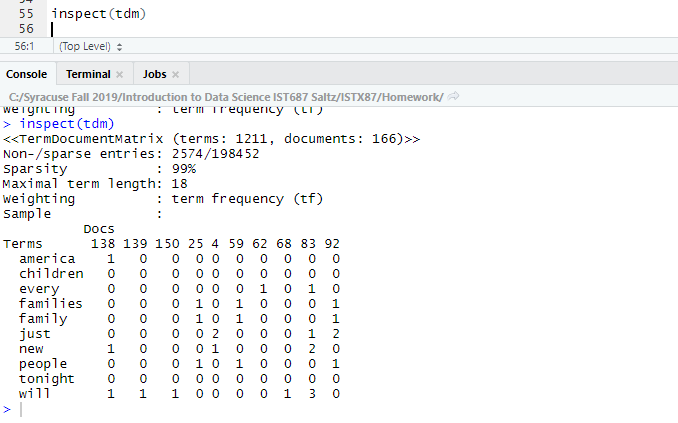


* 1. Create a TermDoumentMatrix variable called ‘tdm’, from the words.corpus variable.



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

The output is of inspect(tdm) the term document matrix shows there are 166 documents and 1211 terms. It returns samples of terms taken randomly from 166 documents and the terms in those documents along with their frequency. It shows the sparsity i.e. how many entries are having zero in the matrix here there are 99% elements which are zero and only 1% non-sparse elements which contain elements other than 0. Also the maximum length of a term in these documents is 18.



1. **List any additional resources that you used here.**
2. **Be sure to save your R file as this will become the starting code for your homework.**

***You must submit all Prep Exercises to blackboard prior to the deadline specified for each assignment.*** PE assignments are due on the evening prior to the lecture class. Late PE assignments will not be accepted for credit.

*Materials provided by the instructors of this course are copyrighted 2018 by Jeffrey Saltz and Jeffrey Stanton, or by the respective instructor that produced them. These materials are provided to enrolled students for their exclusive use during the semester of instruction. Uploading instructor-produced materials to Internet sites (e.g., Coursehero) without permission is considered a breach of professional ethics and will be reported as a possible academic integrity violation.*