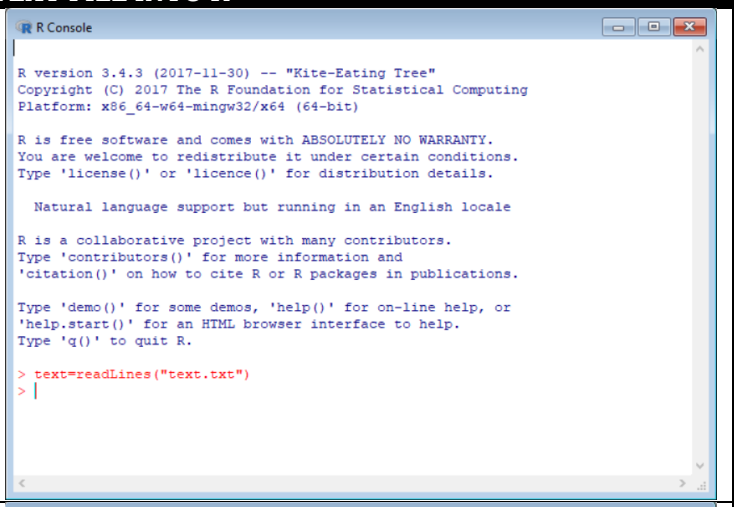
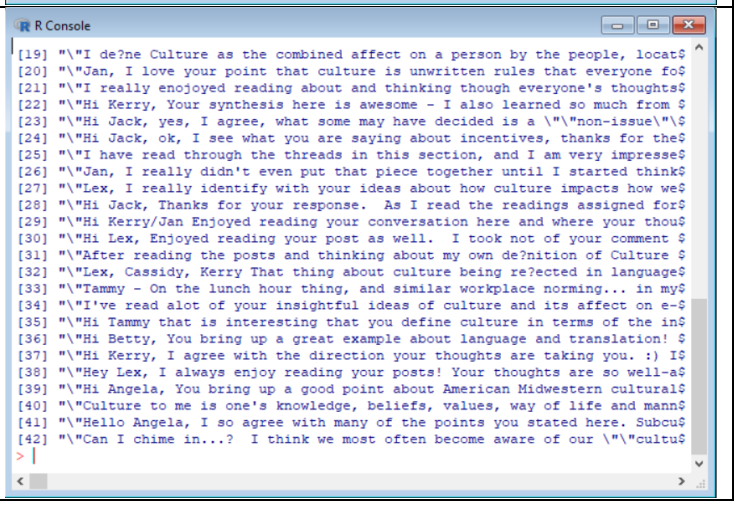
R-Programming Fundamentals for Business Students — Cluster Analysis, Dendrograms, Word Cloud Clusters

Nick V. Flor, University of New Mexico (nickflor@unm.edu)

Assumptions. This tutorial assumes (1) that you had an Excel worksheet with a single column A, where each row represented a social media post (e.g., a tweet), and (2) that you at least removed carriage returns and/or line feeds from each post; and (3) that you saved that file as *text.txt* and, finally, (4) that you ran R and did a *File > Change dir…* to the folder containing *text.txt*

ACTION	REACTION
IMPORTING A TEXT FILE INTO R	
• Type **`text=readLines("text.txt")`** then press **`Enter`** Explanation: This command reads every post into a separate row in the variable **`text`**. If you do not see an error message, you have done this correctly! If you do get an error, retype the command (don't copy and paste)	
• Type **`text`** (or whatever you named your variable) then press **`Enter`** to see your data Explanation: Typing any variable allows you to see its content. Note: The number of rows (e.g., 42) should agree with the number of rows in your Excel spreadsheet. If not, then you did not remove all Enters (carriage returns / line feeds) and you will have to go back and start over.	

(continued on next page)

INSTALL PACKAGES

- If you haven't done so in a previous tutorial, install the following packages:
  `install.packages("tm")`
  `install.packages("SnowballC")`
  `install.packages("wordcloud")`

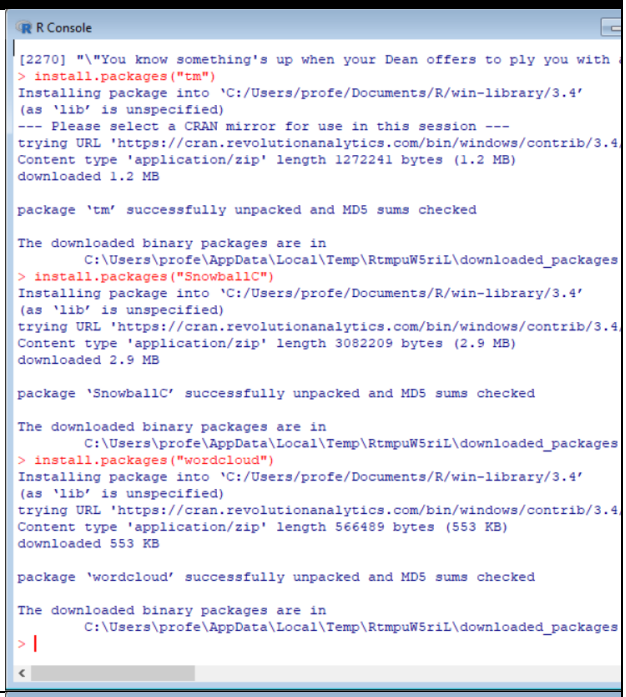
Note: This can take several minutes and freeze your keyboard. Just wait it out.

Explanation:

"tm" is R's text mining package

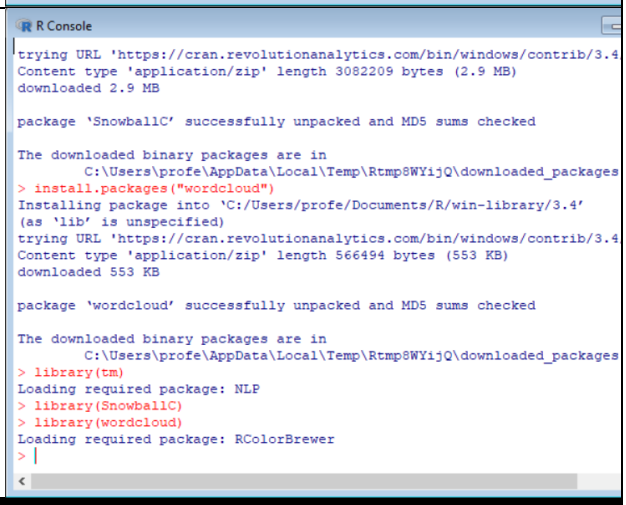
"SnowballC" is a package to stem words (e.g., "turn", "turns", "turning", "turned", all become "turn")

"wordcloud" generates word clouds.



- Type the following:
  `library("tm")`
  `library("SnowballC")`
  `library("wordcloud")`

Explanation: **`install.packages`** merely loads the packages onto your computer. The **`library`** command allows you to use the packages in your current R session.

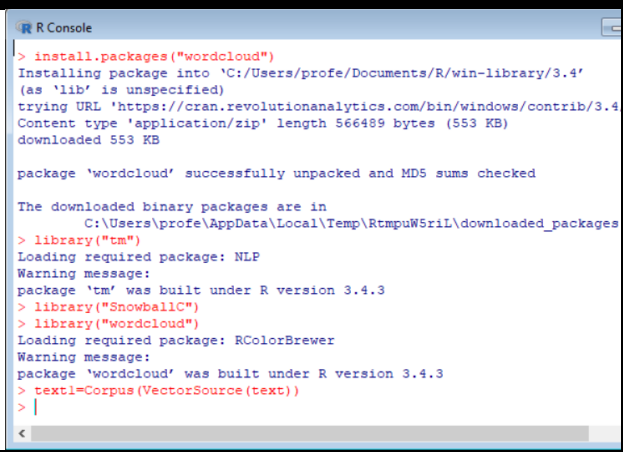


CLEAN DATA (DATA CLEANING)

- Clean the data. One way to do so is to type:

```
text1=tolower(text)
text2=removeWords(text1,stopwords("english"))
text3=gsub("/"," ",text2)
text4=removePunctuation(text3)
text5=stemDocument(text4)
text6=stripWhitespace(text5)
```

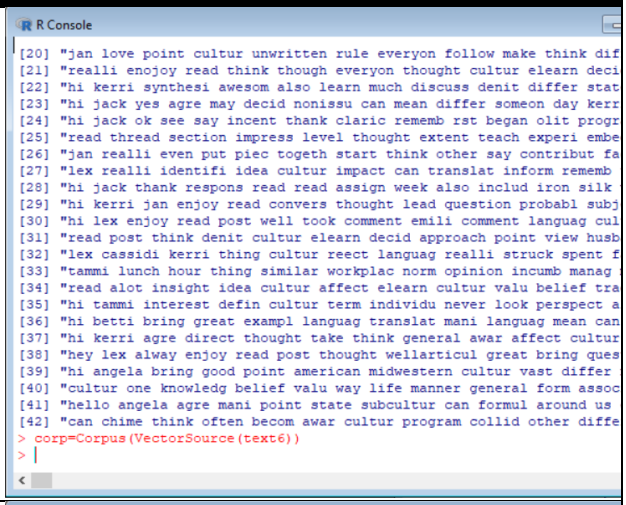
Explanation: Converts the original text to lower case, while removing stopwords, converting slashes to spaces, removing punctuation, stemming the document, and stripping whitespace.



CREATE HIERARCHICAL CLUSTER & PLOT DENDROGRAM

- Type **`corp=Corpus(VectorSource(text6))`**

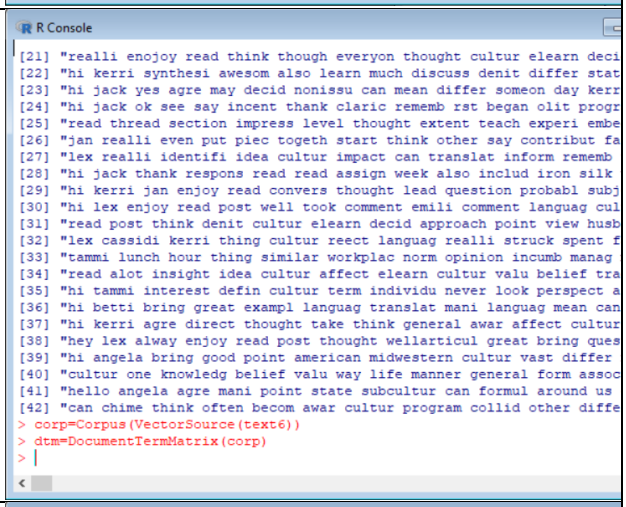
Explanation: a **`Corpus`** is a special data structure which we need to create first before we can create a **`DocumentTermMatrix`**. We store the corpus in the variable **`corp`**.



- Type **`dtm=DocumentTermMatrix(corp)`**

Explanation: The Corpus object (**`corp`** in our example) is converted to a **`DocumentTermMatrix`**, which we store in the variable **`dtm`**. The rows of this matrix correspond to the individual text lines. The columns, however, for each row, correspond to every single word across all text lines.

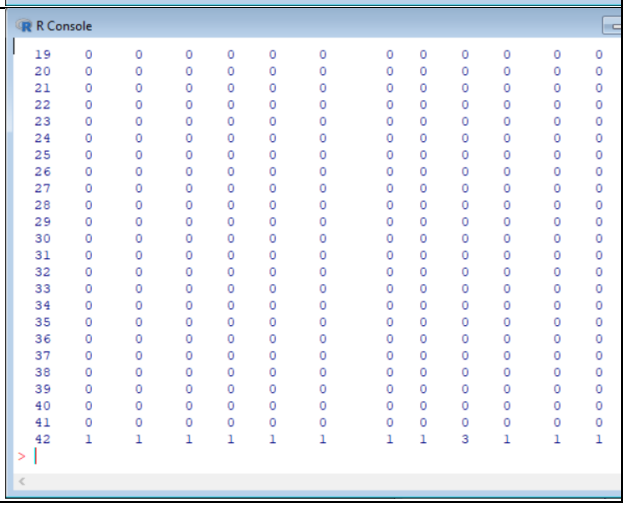
Typing **`dtm`** summarizes the matrix, but does not show you the contents of the matrix.



- Type
  `mat=as.matrix(dtm)`
  `mat`

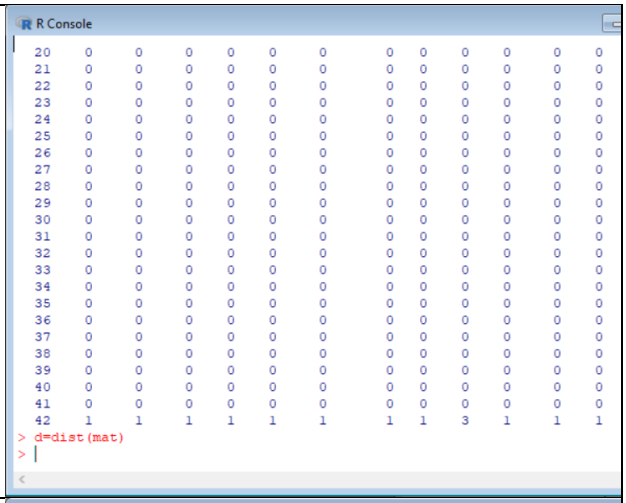
Explanation: **`as.matrix`** converts the **`DocumentTermMatrix`** into a matrix, which we store in **`mat`**, that you can examine directly by typing the variable name. However, the matrix is so big that R doesn't display it properly.

Note: If you want to examine the matrix, you can type **`write.csv(mat, "mat.csv")`** and import the file into Excel (not shown).



- Type **`d=dist(mat)`**

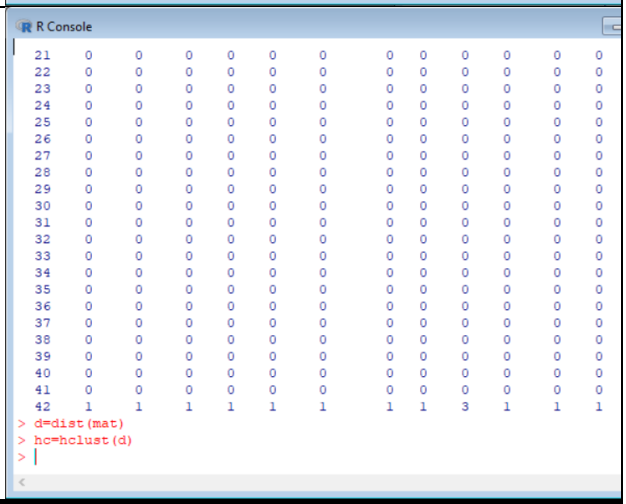
Explanation: The **`dist`** command calculates the similarity between each line of text and every other line of text. Two lines of text that have high overlap between words are said to be "close" in distance. Two lines of text that do not have many words in common are said to be "far" in distance. We store the distances between lines of text in the variable **`d`**.



- Type **`hc=hclust(d)`**

Explanation: The command **`hclust`** clusters our lines of text two lines at a time, starting with the two most similar, then finding the line most similar to those two, then finding the line most similar to those three, and so on. We store the result in a variable **`hc`**.

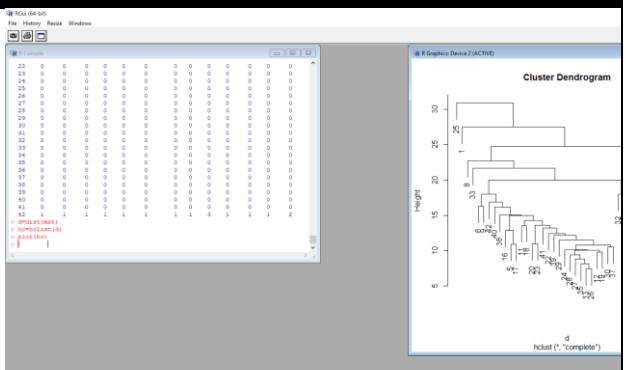
The problem is that we can't see the cluster until we plot it. We do that next.



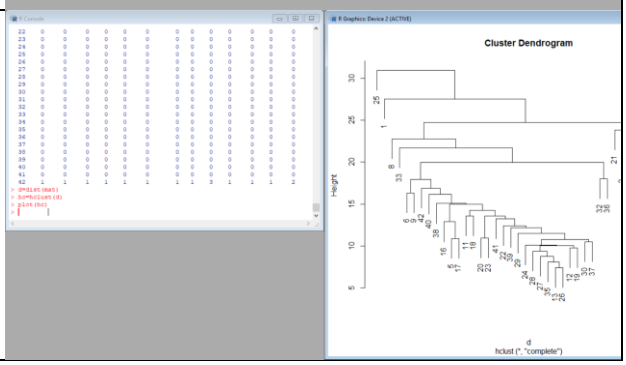
PLOT THE DENDROGRAM

- Type **`plot(hc)`**

Explanation: This plots a tree diagram from the hierarchical cluster (**`hc`**) known as a dendrogram.



- You can resize the dendrogram by resizing the plot window (drag the borders!). This lets you see the clustering a lot better.

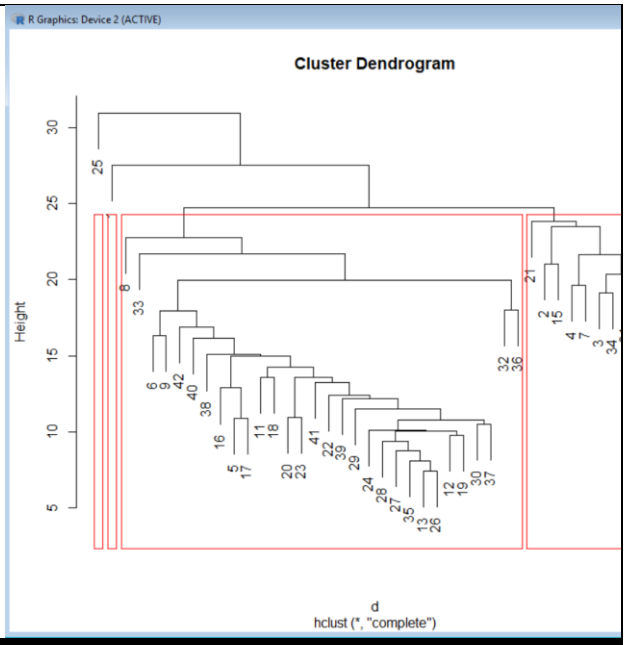


- Type **`rect.hclust(hc, 4)`**

Explanation: You can draw rectangles around clusters using the command **`rect.hclust`**. Experiment with changing the second parameter (4 in this example), to see the various clusters.

Note: When you experiment with the second parameter, you should do a **`plot(hc)`**, prior to the **`rect.hclust`** to clear the previous rectangles.

Note 2: One cluster is in the far right, and consists of lines 21,2,15,4,7,3,34,31,10,14. What is inside the cluster? We can use word clouds to see.



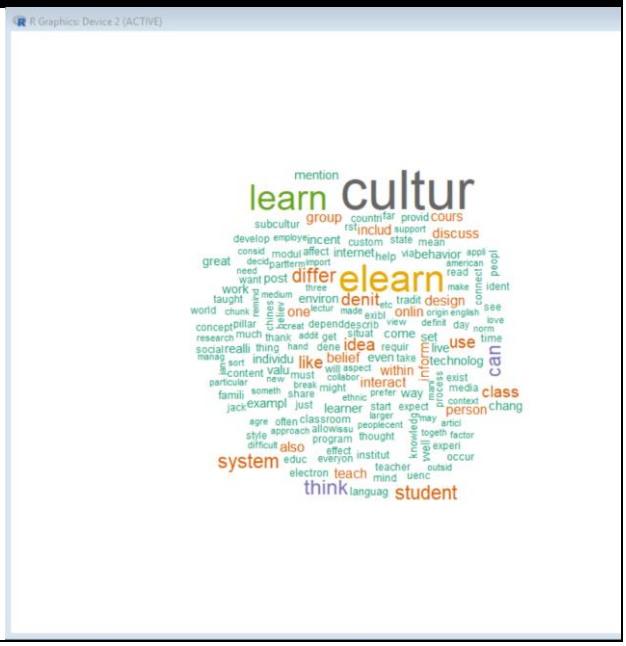
USING WORD CLOUDS TO SEE SPECIFIC CLUSTERS

- Type

`wordcloud(text6[c(21,2,15,4,7,3,34,31,10,14)],col=brewer.pal(8, "Dark2"))`

Explanation: You can easily see the contents of any cluster by doing a word cloud on your text (**`text6`** in this example, but remember that you use whatever the last line was that you data cleaned, e.g., text10 if you did 10 data cleans). The trick to only seeing a certain number of lines is to put those lines in a c-vector **`c(…)`**, then indexing text using that c-vector, e.g., **`textN[c(…)]`**

Note: You may want to experiment with min.freq=1, if your cluster is too small. Or plug a higher number in for min.freq if your cluster is too big.



You're done. Try experimenting with plotting other clusters in the dendrogram, because you know you'll get a question about it in the upcoming exam.