TopicModeling

Sowmya Vajjala

February 8, 2018

Purpose: Demonstrating how to do topic modeling in R with tm and topicmodels packages. Tutorial described in https://eight2late.wordpress.com/2015/09/29/a-gentle-introduction-to-topic-modeling-using-r/ is being used here with some changes accounting for the errors one may face. Note: I am not doing all the pre-processing described in the tutorial. If you want, you can do that.

```
#set your working directory to where all the .txt files in the corpus are.

#install required libraries: tm, topicmodels, snowballC

#load required libraries
library("tm")
```

```
## Loading required package: NLP
```

```
library("topicmodels")

#get listing of .txt files in directory
filenames <- list.files(getwd(),pattern="*.txt")

#read files into a character vector
files <- lapply(filenames,readLines)

#create corpus from vector using Corpus function in tm
docs <- Corpus(VectorSource(files))

#start preprocessing: Transform to lower case, remove stopwords,
#remove numbers, punctuation, strip whitespace, do stemming

#lazy=TRUE seems to be required on some operating systems.
#So, I am putting that in comments - if you need, uncomment that part and adjust parantheses.
my_corpus <- tm_map(docs, content_transformer(tolower))#,lazy=TRUE)
my_corpus <- tm_map(my_corpus, removeWords, stopwords("english"))#, lazy=TRUE)
my_corpus <- tm_map(my_corpus, removeNumbers)#, lazy=TRUE)
my_corpus <- tm_map(my_corpus, removePunctuation)#, lazy=TRUE)
my_corpus <- tm_map(my_corpus, stripWhitespace)#, lazy=TRUE)
my_corpus <- tm_map(my_corpus, stemDocument)#, lazy=TRUE) #Is Stemming needed? Why?

#If you see the following error:
#UseMethod("meta", x) : no applicable method for 'meta' applied to an object of class "try-error"
#Then, uncomment the below line. I did not see this on Linux and MacOS.
#my_corpus <- tm_map(my_corpus, content_transformer(function(x) iconv(x, to='UTF-8-MAC', sub='byte')),

#You may need this on Windows, I am not sure.
#my_corpus <- tm_map(my_corpus, PlainTextDocument)

#Create document-term matrix
myDtm <- DocumentTermMatrix(my_corpus)
```

```r
#convert rownames to filenames
rownames(myDtm) <- filenames

#collapse matrix by summing over columns
freq <- colSums(as.matrix(myDtm))

#length should be total number of terms
length(freq)
```

```
## [1] 6513
```

```r
#List all terms in decreasing order of freq and write to disk
ord <- order(freq,decreasing=TRUE)
#freq[ord]
write.csv(freq[ord],"../word_freq.csv")

#Topic modeling using LDA:

#Set parameters for Gibbs sampling
burnin <- 4000
iter <- 2000
thin <- 500
seed <-list(2003,5,63,100001,765)
nstart <- 5
best <- TRUE

#number of topics
k <- 5

#Run LDA using Gibbs sampling
ldaOut <-LDA(myDtm,k, method="Gibbs", control=list(nstart=nstart, seed = seed,
                                                   best=best, burnin = burnin, iter = iter, thin=thin))

#write out results
#docs to topics
ldaOut.topics <- as.matrix(topics(ldaOut))
write.csv(ldaOut.topics,file=paste("LDAGibbs",k,"DocsToTopics.csv"))

#top 10 terms in each topic
ldaOut.terms <- as.matrix(terms(ldaOut,10))
write.csv(ldaOut.terms,file=paste("LDAGibbs",k,"TopicsToTerms.csv"))

#probabilities associated with each topic assignment
topicProbabilities <- as.data.frame(ldaOut@gamma)
write.csv(topicProbabilities,file=paste("LDAGibbs",k,"TopicProbabilities.csv"))

#Find relative importance of top 2 topics
topic1ToTopic2 <- lapply(1:nrow(myDtm),function(x)
sort(topicProbabilities[x,])[k]/sort(topicProbabilities[x,])[k-1])

#Find relative importance of second and third most important topics
topic2ToTopic3 <- lapply(1:nrow(myDtm),function(x)
sort(topicProbabilities[x,])[k-1]/sort(topicProbabilities[x,])[k-2])
```

```
#write to file
write.csv(topic1ToTopic2,file=paste("LDAGibbs",k,"Topic1ToTopic2.csv"))
write.csv(topic2ToTopic3,file=paste("LDAGibbs",k,"Topic2ToTopic3.csv"))
```