Metadata 2016 Spring

SY YU

2016-03-30 / 2016-03-25

```
#How to Conver R scrip to pdf file.
#Installing pandoc / Miktex FIRST.
#library(rmarkdown)
#render("input.Rmd", c("html_document", "pdf_document"))
```

date: 2016-03-25

```
library(XML)
library(RCurl)
```

```
## Loading required package: bitops
```

```
fileURL <- "http://www.w3schools.com/xml/simple.xml"
doc <- xmlTreeParse(fileURL, useInternal=TRUE)
doc
```

```
## <?xml version="1.0" encoding="UTF-8"?>
## <breakfast_menu>
##   <food>
##     <name>Belgian Waffles</name>
##     <price>$5.95</price>
##     <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##     <calories>650</calories>
##   </food>
##   <food>
##     <name>Strawberry Belgian Waffles</name>
##     <price>$7.95</price>
##     <description>Light Belgian waffles covered with strawberries and whipped cream</description>
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>Berry-Berry Belgian Waffles</name>
##     <price>$8.95</price>
##     <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>French Toast</name>
##     <price>$4.50</price>
##     <description>Thick slices made from our homemade sourdough bread</description>
##     <calories>600</calories>
##   </food>
##   <food>
##     <name>Homestyle Breakfast</name>
##     <price>$6.95</price>
##     <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
```

```
##     <calories>950</calories>
##   </food>
## </breakfast_menu>
##
```

```
rootNode <-xmlRoot(doc)
rootNode
```

```
## <breakfast_menu>
##   <food>
##     <name>Belgian Waffles</name>
##     <price>$5.95</price>
##     <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##     <calories>650</calories>
##   </food>
##   <food>
##     <name>Strawberry Belgian Waffles</name>
##     <price>$7.95</price>
##     <description>Light Belgian waffles covered with strawberries and whipped cream</description>
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>Berry-Berry Belgian Waffles</name>
##     <price>$8.95</price>
##     <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>French Toast</name>
##     <price>$4.50</price>
##     <description>Thick slices made from our homemade sourdough bread</description>
##     <calories>600</calories>
##   </food>
##   <food>
##     <name>Homestyle Breakfast</name>
##     <price>$6.95</price>
##     <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
##     <calories>950</calories>
##   </food>
## </breakfast_menu>
```

```
xmlName(rootNode)
```

```
## [1] "breakfast_menu"
```

```
names(rootNode)
```

```
##   food   food   food   food   food
## "food" "food" "food" "food" "food"
```

```
rootNode[1]
```

```
## $food
## <food>
##   <name>Belgian Waffles</name>
##   <price>$5.95</price>
##   <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##   <calories>650</calories>
## </food>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```
rootNode[[1]]
```

```
## <food>
##   <name>Belgian Waffles</name>
##   <price>$5.95</price>
##   <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##   <calories>650</calories>
## </food>
```

```
rootNode[[1]][1]
```

```
## $name
## <name>Belgian Waffles</name>
##
## attr(,"class")
## [1] "XMLInternalNodeList" "XMLNodeList"
```

```
rootNode[[1]][[1]]
```

```
## <name>Belgian Waffles</name>
```

```
rootNode[[1]][[2]]
```

```
## <price>$5.95</price>
```

```
########extracting metadata element values
### /Top node
### //node at any levels

xmlValue(rootNode[[1]])
```

```
## [1] "Belgian Waffles$5.95Two of our famous Belgian Waffles with plenty of real maple syrup650"
```

```
xmlValue(rootNode[[1]][[1]])
```

```
## [1] "Belgian Waffles"
```

```
xmlSApply(rootNode,xmlValue)
```

```
##
##                                "Belgian Waffles$5.95Two of our famous Belgian Waffles with plenty of
##
##                     "Strawberry Belgian Waffles$7.95Light Belgian waffles covered with strawberries a
##
## "Berry-Berry Belgian Waffles$8.95Light Belgian waffles covered with an assortment of fresh berries a
##
##                                                 "French Toast$4.50Thick slices made from our homemade
##
##                          "Homestyle Breakfast$6.95Two eggs, bacon or sausage, toast, and our ever-popu
```

```
xpathSApply(rootNode,"/breakfast_menu",xmlValue)
```

```
## [1] "Belgian Waffles$5.95Two of our famous Belgian Waffles with plenty of real maple syrup650Strawbe
```

```
xpathSApply(rootNode,"//name",xmlValue)
```

```
## [1] "Belgian Waffles"             "Strawberry Belgian Waffles"
## [3] "Berry-Berry Belgian Waffles" "French Toast"
## [5] "Homestyle Breakfast"
```

```
xpathSApply(rootNode,"//price",xmlValue)
```

```
## [1] "$5.95" "$7.95" "$8.95" "$4.50" "$6.95"
```

date: 2016-04-01

```
#####Continuing
library(XML)
library(RCurl)
fileURL <- "http://www.w3schools.com/xml/simple.xml"
doc <- xmlTreeParse(fileURL, useInternal=TRUE)
doc
```

```
## <?xml version="1.0" encoding="UTF-8"?>
## <breakfast_menu>
##   <food>
##     <name>Belgian Waffles</name>
##     <price>$5.95</price>
##     <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##     <calories>650</calories>
##   </food>
##   <food>
##     <name>Strawberry Belgian Waffles</name>
##     <price>$7.95</price>
##     <description>Light Belgian waffles covered with strawberries and whipped cream</description>
##     <calories>900</calories>
##   </food>
```

```
##   <food>
##     <name>Berry-Berry Belgian Waffles</name>
##     <price>$8.95</price>
##     <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>French Toast</name>
##     <price>$4.50</price>
##     <description>Thick slices made from our homemade sourdough bread</description>
##     <calories>600</calories>
##   </food>
##   <food>
##     <name>Homestyle Breakfast</name>
##     <price>$6.95</price>
##     <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
##     <calories>950</calories>
##   </food>
## </breakfast_menu>
##
```

```
rootNode <-xmlRoot(doc)

rootNode
```

```
## <breakfast_menu>
##   <food>
##     <name>Belgian Waffles</name>
##     <price>$5.95</price>
##     <description>Two of our famous Belgian Waffles with plenty of real maple syrup</description>
##     <calories>650</calories>
##   </food>
##   <food>
##     <name>Strawberry Belgian Waffles</name>
##     <price>$7.95</price>
##     <description>Light Belgian waffles covered with strawberries and whipped cream</description>
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>Berry-Berry Belgian Waffles</name>
##     <price>$8.95</price>
##     <description>Light Belgian waffles covered with an assortment of fresh berries and whipped cream
##     <calories>900</calories>
##   </food>
##   <food>
##     <name>French Toast</name>
##     <price>$4.50</price>
##     <description>Thick slices made from our homemade sourdough bread</description>
##     <calories>600</calories>
##   </food>
##   <food>
##     <name>Homestyle Breakfast</name>
##     <price>$6.95</price>
##     <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
```

```
##     <calories>950</calories>
##   </food>
## </breakfast_menu>
```

```
xmlName(rootNode)
```

```
## [1] "breakfast_menu"
```

```
menu_all <- xpathSApply(rootNode,"/breakfast_menu",xmlValue)
menu_name <- xpathSApply(rootNode,"//name",xmlValue)
menu_desc <- xpathSApply(rootNode,"//description",xmlValue)

#####Text mining 101
library(tm)
library(SnowballC)
```

```
## Warning: package 'SnowballC' was built under R version 3.2.3
```

```
###How to convert vector of characters to corpus input for the DocumentTermMatrix function from tm packa
#####http://stackoverflow.com/questions/29209873/how-to-convert-vector-of-characters-to-corpus-input-fo
###NLP 101
#####https://rstudio-pubs-static.s3.amazonaws.com/31867_8236987cf0a8444e962ccd2aec46d9c3.html

###Converting object type
class(menu_desc)
```

```
## [1] "character"
```

```
docs <- Corpus(VectorSource(menu_desc))
class(docs)
```

```
## [1] "VCorpus" "Corpus"  "list"
```

```
###Pre-processing
docs <- tm_map(docs, tolower)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeWords, stopwords("english"))

###Tokenizing
strsplit_space_tokenizer <- function(x)
  unlist(strsplit(as.character(x), "[[:space:]]+"))
token_docs<-(sapply(docs, strsplit_space_tokenizer))
#token_docs<-(sapply(docs$content, strsplit_space_tokenizer))

###Stemming
stem_docs <- sapply(token_docs, stemDocument)

###Lemmatization
#http://stackoverflow.com/questions/22993796/lemmatizer-in-r-or-python-am-are-is-be
#name="corpusConfig"
#value="eme" Early Modern English
```

```
#value="ece" Eighteen Century English
#value="ncf" Nineteenth Century Fiction
library(httr)

lemmatize <- function(wordlist) {
  get.lemma <- function(word, url) {
    response <- GET(url,query=list(spelling=word,standardize="",
                                   wordClass="",wordClass2="",
                                   corpusConfig="eme",    # Early Modern English
                                   media="xml"
                                   ))
    content <- content(response,type="text", encoding="UTF-8")
    xml     <- xmlInternalTreeParse(content)
    return(xmlValue(xml["//lemma"][[1]]))
  }
  require(httr)
  require(XML)
  url <- "http://devadorner.northwestern.edu/maserver/lemmatizer"
  return(sapply(wordlist,get.lemma,url=url))
}

###for example,
lemmatize("waffl")
```

```
##   waffl
## "waffl"
```

```
lemmatize("waffles")
```

```
##  waffles
## "waffle"
```

```
lemma_docs <- sapply(token_docs, lemmatize)

###Term-Doc Matrix with Lemmatization
lemma_docs <- Corpus(VectorSource(lemma_docs))
tdm <- TermDocumentMatrix(lemma_docs,
                          control = list(removePunctuation = TRUE,
                                         weighting=weightTfIdf,
                                         stopwords = TRUE))

inspect(tdm[1,])
```

```
## A term-document matrix (1 terms, 5 documents)
##
## Non-/sparse entries: 1/4
## Sparsity           : 80%
## Maximal term length: 10
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##             Docs
## Terms        1 2        3 4 5
##   assortment 0 0 0.257992 0 0
```

```
inspect(tdm[1:29,])
```

```
## A term-document matrix (29 terms, 5 documents)
##
## Non-/sparse entries: 38/107
## Sparsity           : 74%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##              Docs
## Terms                 1         2          3        4        5
##   assortment  0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bacon       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   belgian     0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   berry       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bread       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   brown       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   cover       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   cream       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   egg         0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   everpopular 0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   famou       0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   fresh       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   hash        0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   homemade    0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   light       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   make        0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   maple       0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   plenty      0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   real        0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   sausage     0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   slice       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   sourdough   0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   strawberry  0.0000000 0.3317040 0.00000000 0.000000 0.000000
##   syrup       0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   thick       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   toast       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   two         0.1652410 0.0000000 0.00000000 0.000000 0.165241
##   waffle      0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   whip        0.0000000 0.1888469 0.14688090 0.000000 0.000000
```

```
inspect(tdm[1:10,])
```

```
## A term-document matrix (10 terms, 5 documents)
##
## Non-/sparse entries: 14/36
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##              Docs
## Terms                 1         2          3        4        5
```

```
##   assortment  0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bacon       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   belgian     0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   berry       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bread       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   brown       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   cover       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   cream       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   egg         0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   everpopular 0.0000000 0.0000000 0.00000000 0.000000 0.290241
```

```
###Doc-Term Matrix with Lemmatization
dtm <- DocumentTermMatrix(lemma_docs,
                          control = list(removePunctuation = TRUE,
                                         weighting =function(x)weightTfIdf(x, normalize = FALSE),
                                         stopwords = TRUE))

inspect(dtm[1,])
```

```
## A document-term matrix (1 documents, 29 terms)
##
## Non-/sparse entries: 8/21
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assortment bacon   belgian berry bread brown cover cream egg
##    1          0     0 0.7369656     0     0     0     0     0   0
##     Terms
## Docs everpopular    famou fresh hash homemade light make    maple   plenty
##    1           0 2.321928     0    0        0     0    0 2.321928 2.321928
##     Terms
## Docs     real sausage slice sourdough strawberry    syrup thick toast
##    1 2.321928       0     0         0          0 2.321928     0     0
##     Terms
## Docs      two    waffle whip
##    1 1.321928 0.7369656    0
```

```
inspect(dtm[1,1:10])
```

```
## A document-term matrix (1 documents, 10 terms)
##
## Non-/sparse entries: 1/9
## Sparsity           : 90%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assortment bacon   belgian berry bread brown cover cream egg
##    1          0     0 0.7369656     0     0     0     0     0   0
##     Terms
## Docs everpopular
##    1           0
```

```r
inspect(dtm[1,1:29])
```

```
## A document-term matrix (1 documents, 29 terms)
##
## Non-/sparse entries: 8/21
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assortment bacon   belgian berry bread brown cover cream egg
##    1          0     0 0.7369656     0     0     0     0     0   0
##     Terms
## Docs everpopular    famou fresh hash homemade light make    maple   plenty
##    1           0 2.321928     0    0        0     0    0 2.321928 2.321928
##     Terms
## Docs     real sausage slice sourdough strawberry    syrup thick toast
##    1 2.321928       0     0         0          0 2.321928     0     0
##     Terms
## Docs      two    waffle whip
##    1 1.321928 0.7369656    0
```

```r
###Compare & Recommend with Lemmatization
findAssocs(dtm, "toast", corlimit=0.1)
```

```
##             toast
## bacon        1.00
## brown        1.00
## egg          1.00
## everpopular  1.00
## hash         1.00
## sausage      1.00
## two          0.61
```

```r
findAssocs(dtm, "waffle", corlimit=0.1)
```

```
##            waffle
## belgian      1.00
## cover        0.67
## cream        0.67
## light        0.67
## whip         0.67
## assortment   0.41
## berry        0.41
## famou        0.41
## fresh        0.41
## maple        0.41
## plenty       0.41
## real         0.41
## strawberry   0.41
## syrup        0.41
```

```
menu_rec<-findAssocs(dtm, c("toast", "waffle"), corlimit=0.5)
menu_rec
```

```
## $toast
##       bacon       brown         egg everpopular        hash     sausage
##        1.00        1.00        1.00        1.00        1.00        1.00
##         two
##        0.61
##
## $waffle
## belgian   cover   cream   light    whip
##    1.00    0.67    0.67    0.67    0.67
```

```
menu_rec<-findAssocs(dtm, c("berry", "egg"), corlimit=0.1)
menu_rec
```

```
## $berry
## assortment      fresh      cover      cream      light       whip
##       1.00       1.00       0.61       0.61       0.61       0.61
##    belgian     waffle
##       0.41       0.41
##
## $egg
##       bacon       brown everpopular        hash     sausage       toast
##        1.00        1.00        1.00        1.00        1.00        1.00
##         two
##        0.61
```

```
menu_rec<-findAssocs(dtm, c("sausage", "egg"), corlimit=0.1)
menu_rec
```

```
##             sausage  egg
## bacon          1.00 1.00
## brown          1.00 1.00
## everpopular    1.00 1.00
## hash           1.00 1.00
## toast          1.00 1.00
## two            0.61 0.61
```

```
menu_rec<-findAssocs(dtm, c("sausage", "egg"), corlimit=0.5)
menu_rec
```

```
##             sausage  egg
## bacon          1.00 1.00
## brown          1.00 1.00
## everpopular    1.00 1.00
## hash           1.00 1.00
## toast          1.00 1.00
## two            0.61 0.61
```

```
### With Stemmed docs

###Term-Doc Matrix with Stemming
class(stem_docs)
```

```
## [1] "list"
```

```
stem_docs <- Corpus(VectorSource(stem_docs))
class(stem_docs)
```

```
## [1] "VCorpus" "Corpus"  "list"
```

```
tdm <- TermDocumentMatrix(stem_docs,
                          control = list(removePunctuation = TRUE,
                                         weighting=weightTfIdf,
                                         stopwords = TRUE))

inspect(tdm[1,])
```

```
## A term-document matrix (1 terms, 5 documents)
##
## Non-/sparse entries: 1/4
## Sparsity           : 80%
## Maximal term length: 6
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##         Docs
## Terms    1 2        3 4 5
##   assort 0 0 0.257992 0 0
```

```
inspect(tdm[1:29,])
```

```
## A term-document matrix (29 terms, 5 documents)
##
## Non-/sparse entries: 38/107
## Sparsity           : 74%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##             Docs
## Terms                1         2          3        4        5
##   assort      0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bacon       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   belgian     0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   berri       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bread       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   brown       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   cover       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   cream       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   egg         0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   everpopular 0.0000000 0.0000000 0.00000000 0.000000 0.290241
```

```
##   famous      0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   fresh       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   hash        0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   homemad     0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   light       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   made        0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   mapl        0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   plenti      0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   real        0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   sausag      0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   slice       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   sourdough   0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   strawberri  0.0000000 0.3317040 0.00000000 0.000000 0.000000
##   syrup       0.2902410 0.0000000 0.00000000 0.000000 0.000000
##   thick       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   toast       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   two         0.1652410 0.0000000 0.00000000 0.000000 0.165241
##   waffl       0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   whip        0.0000000 0.1888469 0.14688090 0.000000 0.000000
```

```
inspect(tdm[1:10,])
```

```
## A term-document matrix (10 terms, 5 documents)
##
## Non-/sparse entries: 14/36
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
##
##              Docs
## Terms                 1         2          3        4        5
##   assort      0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bacon       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   belgian     0.0921207 0.1052808 0.08188507 0.000000 0.000000
##   berri       0.0000000 0.0000000 0.25799201 0.000000 0.000000
##   bread       0.0000000 0.0000000 0.00000000 0.386988 0.000000
##   brown       0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   cover       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   cream       0.0000000 0.1888469 0.14688090 0.000000 0.000000
##   egg         0.0000000 0.0000000 0.00000000 0.000000 0.290241
##   everpopular 0.0000000 0.0000000 0.00000000 0.000000 0.290241
```

```
###Doc-Term Matrix with Stemming
dtm <- DocumentTermMatrix(stem_docs,
                          control = list(removePunctuation = TRUE,
                                         weighting =function(x)weightTfIdf(x, normalize = FALSE),
                                         stopwords = TRUE))

inspect(dtm[1,])
```

```
## A document-term matrix (1 documents, 29 terms)
##
## Non-/sparse entries: 8/21
```

```
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assort bacon   belgian berri bread brown cover cream egg everpopular
##    1      0     0 0.7369656     0     0     0     0     0   0           0
##     Terms
## Docs   famous fresh hash homemad light made     mapl   plenti     real
##    1 2.321928     0    0       0     0    0 2.321928 2.321928 2.321928
##     Terms
## Docs sausag slice sourdough strawberri    syrup thick toast      two
##    1      0     0         0          0 2.321928     0     0 1.321928
##     Terms
## Docs     waffl whip
##    1 0.7369656    0
```

```
inspect(dtm[1,1:10])
```

```
## A document-term matrix (1 documents, 10 terms)
##
## Non-/sparse entries: 1/9
## Sparsity           : 90%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assort bacon   belgian berri bread brown cover cream egg everpopular
##    1      0     0 0.7369656     0     0     0     0     0   0           0
```

```
inspect(dtm[1,1:29])
```

```
## A document-term matrix (1 documents, 29 terms)
##
## Non-/sparse entries: 8/21
## Sparsity           : 72%
## Maximal term length: 11
## Weighting          : term frequency - inverse document frequency (tf-idf)
##
##     Terms
## Docs assort bacon   belgian berri bread brown cover cream egg everpopular
##    1      0     0 0.7369656     0     0     0     0     0   0           0
##     Terms
## Docs   famous fresh hash homemad light made     mapl   plenti     real
##    1 2.321928     0    0       0     0    0 2.321928 2.321928 2.321928
##     Terms
## Docs sausag slice sourdough strawberri    syrup thick toast      two
##    1      0     0         0          0 2.321928     0     0 1.321928
##     Terms
## Docs     waffl whip
##    1 0.7369656    0
```

```
###Compare & Recommend with Stemming
findAssocs(dtm, "toast", corlimit=0.1)
```

```
##             toast
## bacon        1.00
## brown        1.00
## egg          1.00
## everpopular  1.00
## hash         1.00
## sausag       1.00
## two          0.61
```

```
findAssocs(dtm, "waffl", corlimit=0.1)
```

```
##            waffl
## belgian     1.00
## cover       0.67
## cream       0.67
## light       0.67
## whip        0.67
## assort      0.41
## berri       0.41
## famous      0.41
## fresh       0.41
## mapl        0.41
## plenti      0.41
## real        0.41
## strawberri  0.41
## syrup       0.41
```

```
menu_rec<-findAssocs(dtm, c("toast", "waffl"), corlimit=0.5)
menu_rec
```

```
## $toast
##       bacon       brown         egg everpopular        hash      sausag
##        1.00        1.00        1.00        1.00        1.00        1.00
##         two
##        0.61
##
## $waffl
## belgian   cover   cream   light    whip
##    1.00    0.67    0.67    0.67    0.67
```

```
menu_rec<-findAssocs(dtm, c("berri", "egg"), corlimit=0.1)
menu_rec
```

```
## $berri
##  assort   fresh   cover   cream   light    whip belgian   waffl
##    1.00    1.00    0.61    0.61    0.61    0.61    0.41    0.41
##
## $egg
```

```
##       bacon       brown everpopular        hash      sausag       toast
##        1.00        1.00        1.00        1.00        1.00        1.00
##         two
##        0.61
```

```
menu_rec<-findAssocs(dtm, c("sausag", "egg"), corlimit=0.1)
menu_rec
```

```
##             sausag  egg
## bacon         1.00 1.00
## brown         1.00 1.00
## everpopular   1.00 1.00
## hash          1.00 1.00
## toast         1.00 1.00
## two           0.61 0.61
```

```
menu_rec<-findAssocs(dtm, c("sausag", "egg"), corlimit=0.5)
menu_rec
```

```
##             sausag  egg
## bacon         1.00 1.00
## brown         1.00 1.00
## everpopular   1.00 1.00
## hash          1.00 1.00
## toast         1.00 1.00
## two           0.61 0.61
```