Code ▾

Text Mining Pipeline - N-gram

A simple R project adapted for self-study purpose.

In this project, we perform analysis on a customer complaint dataset with N-gram language model using R.

Outline

- Section 1: Load Packages
- Section 2: Load Data
- Section 3: Text Cleaning
- Section 4: N-Gram Tokenization
- Section 5: Graph Plotting for N-Gram

Section 1: Load Packages

The R packages used in this project are listed below:

- `tm` (https://cran.r-project.org/web/packages/tm/index.html): a **text mining** package which provides many functions for text mining applications in R.
- `RWeka` (https://cran.r-project.org/web/packages/RWeka/index.html): provides machine learning algorithms for R applications.

Hide

```
library(tm)
library(RWeka)
```

```
package 㤼㸱RWeka㤼㸲 was built under R version 3.6.3
```

Section 2: Load Data

The dataset we use in this project is the consumer complaint dataset (https://catalog.data.gov/dataset/consumer-complaint-database), which consists of 18 features (columns). For our purpose, we will only use a very small subset of the original dataset (199 records out of **3 million records from the original dataset!**)

First, read the dataset from a CSV file into a data frame.

Hide

```
#read data from csv into dataframe
# complaintData <- read.csv("C:/r/simpleTextMining/data/complaintFinance/consumer_complaint_2
00.csv")
complaintData <- read.csv("../data/complaintFinance/consumer_complaint_200.csv")

#use head function to inspect first 6 rows of data
head(complaintData)
```

	Date.received <fctr>	Product <fctr>
1	5/10/2019	Checking or savings account
2	5/10/2019	Checking or savings account
3	5/10/2019	Debt collection
4	5/10/2019	Credit reporting, credit repair services, or other personal consumer reports
5	5/10/2019	Checking or savings account
6	5/10/2019	Mortgage

6 rows | 1-3 of 18 columns

Next, read the data from data frame into `VCorpus` . Here, we use only the column: **Issue** from the data frame. Feel free to play around with other columns on you own.

Then, inspect the internal structure of the `VCorpus` by calling `str()` . Note that this function will print the internal structure for all 199 records by default. Therefore, we define `indices` which limits the function to print only the internal structure of first 3 records from `Vcorpus` .

Hide

```
#read data from dataframe into VCorpus
#Use xxx$Issue to read data from Issue column
issueData <- VCorpus(VectorSource(complaintData$Issue))

indices <- seq(1,3)
str(issueData[indices])
```

```
List of 3
 $ 1:List of 2
  ..$ content: chr "Managing an account"
  ..$ meta   :List of 7
  .. ..$ author       : chr(0) 
  .. ..$ datetimestamp: POSIXlt[1:1], format: "2020-10-29 13:07:47"
  .. ..$ description  : chr(0) 
  .. ..$ heading      : chr(0) 
  .. ..$ id           : chr "1"
  .. ..$ language     : chr "en"
  .. ..$ origin       : chr(0) 
  .. ..- attr(*, "class")= chr "TextDocumentMeta"
  ..- attr(*, "class")= chr [1:2] "PlainTextDocument" "TextDocument"
 $ 2:List of 2
  ..$ content: chr "Managing an account"
  ..$ meta   :List of 7
  .. ..$ author       : chr(0) 
  .. ..$ datetimestamp: POSIXlt[1:1], format: "2020-10-29 13:07:47"
  .. ..$ description  : chr(0) 
  .. ..$ heading      : chr(0) 
  .. ..$ id           : chr "2"
  .. ..$ language     : chr "en"
  .. ..$ origin       : chr(0) 
  .. ..- attr(*, "class")= chr "TextDocumentMeta"
  ..- attr(*, "class")= chr [1:2] "PlainTextDocument" "TextDocument"
 $ 3:List of 2
  ..$ content: chr "Communication tactics"
  ..$ meta   :List of 7
  .. ..$ author       : chr(0) 
  .. ..$ datetimestamp: POSIXlt[1:1], format: "2020-10-29 13:07:47"
  .. ..$ description  : chr(0) 
  .. ..$ heading      : chr(0) 
  .. ..$ id           : chr "3"
  .. ..$ language     : chr "en"
  .. ..$ origin       : chr(0) 
  .. ..- attr(*, "class")= chr "TextDocumentMeta"
  ..- attr(*, "class")= chr [1:2] "PlainTextDocument" "TextDocument"
 - attr(*, "class")= chr [1:2] "VCorpus" "Corpus"
```

There are more ways to inspect the data, as demonstrated below:

Hide

```
#view summary of corpus information
summary(issueData[indices])
```

```
  Length Class             Mode
1 2      PlainTextDocument list
2 2      PlainTextDocument list
3 2      PlainTextDocument list
```

Hide

```


#inspect data for all records
inspect(issueData[indices])
```

```
<<VCorpus>>
Metadata:  corpus specific: 0, document level (indexed): 0
Content:  documents: 3

[[1]]
<<PlainTextDocument>>
Metadata:  7
Content:  chars: 19

[[2]]
<<PlainTextDocument>>
Metadata:  7
Content:  chars: 19

[[3]]
<<PlainTextDocument>>
Metadata:  7
Content:  chars: 21
```

Hide

```
#inspect a particular document (e.g. the 1st doc)
writeLines(as.character(issueData[[1]]))
```

```
Managing an account
```

Section 3: Text Cleaning

Perform some transformations and pre-processing on the original text. The steps are demonstrated as follows.

Hide

```
#remove stopwords using the standard list in tm
issueData <- tm_map(issueData, removeWords, stopwords("english"))

#apply removePunctuation in corpus
issueData <- tm_map(issueData, removePunctuation)

#convert corpus to lower case
issueData <- tm_map(issueData, content_transformer(tolower))

#remove digits in corpus
issueData <- tm_map(issueData, removeNumbers)

#remove whitespace (optional to remove extra whitespace)
issueData <- tm_map(issueData, stripWhitespace)
```

Section 4: N-Gram Tokenization

In this section, we create **trigram** tokenizer and use it to create a **term document matrix**.

Hide

```
#create Trigram Tokenizer
TrigramTokenizer <- function(x) RWeka::NGramTokenizer(x, RWeka::Weka_control(min = 3, max = 3
))

#create term document matrix using ngram tokenizer
tdmIssue <- TermDocumentMatrix(issueData, control = list(tokenize = TrigramTokenizer)) # crea
te tdm from n-grams

#inspect summary of term document matrix
tdmIssue
```

```
<<TermDocumentMatrix (terms: 60, documents: 199)>>
Non-/sparse entries: 334/11606
Sparsity           : 97%
Maximal term length: 32
Weighting          : term frequency (tf)
```

Then, get the frequent trigram terms (which appears at least 5 times in the term document matrix).

Hide

```
#get frequent trigram terms (e.g. 5)
tdmIssue.freqtrigram <- findFreqTerms(tdmIssue,lowfreq = 5)

tdmIssue.freqtrigram
```

```
 [1] "attempts collect debt"            "collect debt owed"                "companys investig
ation existing"
 [4] "credit reporting companys"        "false statements representation"  "improper use repo
rt"
 [7] "incorrect information report"     "investigation existing problem"   "negative legal ac
tion"
[10] "problem credit reporting"         "reporting companys investigation" "take negative leg
al"
[13] "threatened take negative"         "took threatened take"             "written notificat
ion debt"
```

Section 5: Graph Plotting for N-Gram

We can manipulate the term document matrix to obtain the frequency of all trigrams generated in previous section.

Hide

```
IssueTrigramfreq <- rowSums(as.matrix(tdmIssue[tdmIssue.freqtrigram,]))
IssueTrigramfreq <- data.frame(word=names(IssueTrigramfreq),frequency=IssueTrigramfreq)

#check the first n items
head(IssueTrigramfreq)
```

	word <fctr>	frequency <dbl>
attempts collect debt	attempts collect debt	56
collect debt owed	collect debt owed	56
companys investigation existing	companys investigation existing	11
credit reporting companys	credit reporting companys	11
false statements representation	false statements representation	7
improper use report	improper use report	7

6 rows

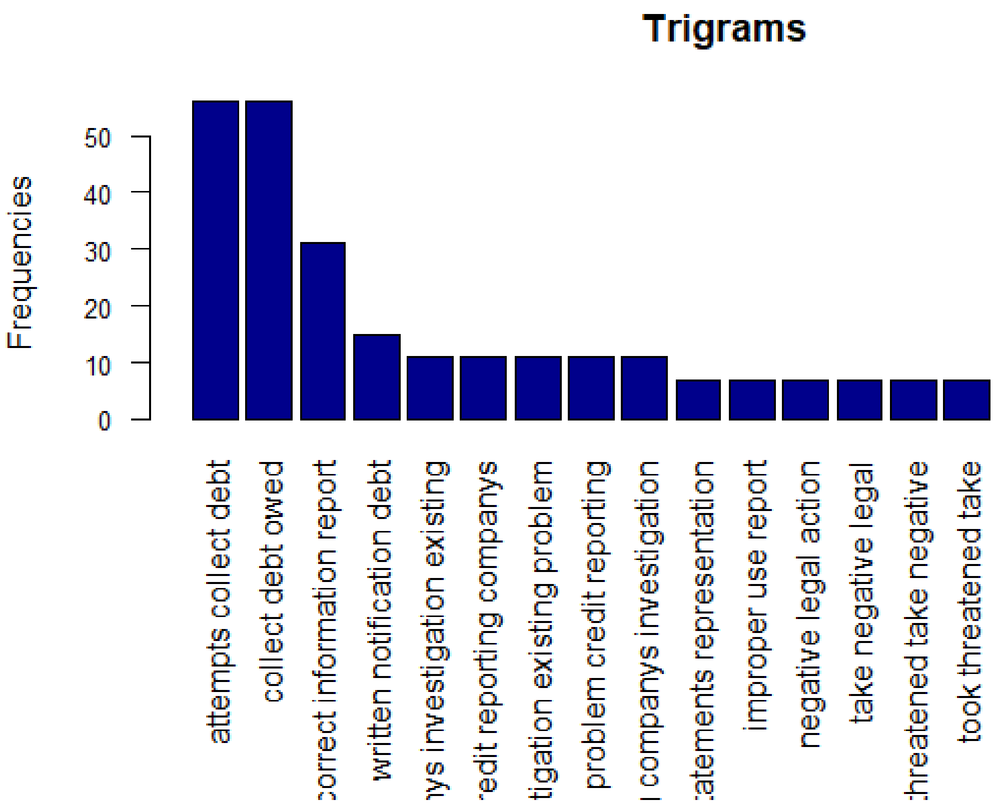
Finally, create a histogram to visualize the trigram in order (from the most frequent to the least frequent).

Hide

```
#create the graph plotting fucntion

plotthegraph <- function(data,title,num){
  df <- data[order(-data$frequency),][1:num,]
  barplot(df[1:num,]$freq, las = 2, names.arg = df[1:num,]$word,
          col ="darkblue", main = title,
          ylab = "Frequencies",cex.axis =0.8)
}
par(mar=c(10,4,4,2))

#plot the graph for Trigram
plotthegraph(IssueTrigramfreq,"Trigrams",20)
```


Trigrams
Frequencies
attempts collect debt
collect debt owed
orrect information report
written notification debt
ys investigation existing
edit reporting companys
tigation existing problem
problem credit reporting
companys investigation
atements representation
improper use report
negative legal action
take negative legal
hreatened take negative
took threatened take