Topic Classification Using Text Mining and Machine Learning

Nahel Rifai

4/20/2017

A project that uses machine learning to classify breaking news. We used a data base of +20,000 news articles from CNBC. Using text mining, we create a model that breaks these news articles into 10 different topics (by seeing how these articles relate in terms of their text). We then assign a category to each of these topics ("Politics", "Tech", "Finance", etc). We finally put the model into use by going into the CNBC website and scraping breaking news (uncategorized). Our model then classifies these articles into one of the aforementioned 10 topics.

Libraries and setting the working directory

```
setwd("~/Desktop/Prasanna/HW2")
library(tm)
```

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

```
library(topicmodels)
library(rvest)
```

```
## Loading required package: xml2
```

```
library(stringr)
library(knitr)
library(png)
```

1. Fit a Topic Model to the existing news data archive

```
archival_data = read.csv("NewsArticles.csv", header = TRUE)
```

Loading and cleaning up the corpus

```
corp.original <- VCorpus(VectorSource(archival_data$content[1:3000])) ##limiting it to 3000 documents
corp <- tm_map(corp.original, removePunctuation)
corp <- tm_map(corp, removeNumbers)
corp <- tm_map(corp, content_transformer(removeWords), c("TIL")  ,lazy=TRUE)
corp <- tm_map(corp, content_transformer(removeWords), stopwords("SMART"),lazy=TRUE)
corp <- tm_map(corp, content_transformer(tolower) ,lazy=TRUE)
corp <- tm_map(corp, content_transformer(stemDocument) ,lazy=TRUE)
myStopwords <- c("percent", "plan", "year", "read", "compani", "inform", "time", "day", "make", "the",
corp <- tm_map(corp, removeWords, myStopwords)
corp <- tm_map(corp, stripWhitespace)
```

Creating the document term matrix

```
dtm = DocumentTermMatrix(corp)
```

Running the LDA model with 10 topics

```
k <- 10
seed <-list(2003,5,63,100001,765, 831, 0101, 2736, 9981, 8)
ldaOut <-LDA(dtm,k, seed = seed)
```

```r
lda.terms <- as.matrix(terms(ldaOut,10)) ##top ten terms per topic
lda.terms
```

```
##       Topic 1   Topic 2   Topic 3      Topic 4   Topic 5   Topic 6
##  [1,] "rate"    "loan"    "obama"      "share"   "busi"    "home"
##  [2,] "price"   "water"   "presid"     "billion" "servic"  "sale"
##  [3,] "oil"     "credit"  "republican" "quarter" "peopl"   "retail"
##  [4,] "fed"     "vehicl"  "state"      "earn"    "million" "price"
##  [5,] "market"  "student" "hous"       "revenu"  "product" "store"
##  [6,] "growth"  "car"     "tax"        "million" "game"    "hous"
##  [7,] "increas" "million" "senat"      "expect"  "googl"   "citi"
##  [8,] "economi" "busi"    "govern"     "analyst" "work"    "market"
##  [9,] "expect"  "secur"   "democrat"   "sale"    "market"  "mortgag"
## [10,] "job"     "consum"  "congress"   "apple"   "content" "weather"
##       Topic 7   Topic 8    Topic 9    Topic 10
##  [1,] "peopl"   "drug"     "state"    "fund"
##  [2,] "retir"   "peopl"    "tax"      "bank"
##  [3,] "insur"   "work"     "ticket"   "market"
##  [4,] "financi" "vaccin"   "airlin"   "stock"
##  [5,] "health"  "develop"  "law"      "investor"
##  [6,] "save"    "patient"  "york"     "invest"
##  [7,] "cost"    "research" "investig" "manag"
##  [8,] "pay"     "user"     "court"    "bond"
##  [9,] "tax"     "health"   "million"  "capit"
## [10,] "money"   "show"     "flight"   "billion"
```

Naming the topics

```r
colnames(lda.terms)[] <- c("Public Policy", "Politics", "Consumer Finance/Debt",
                           "Energy", "Tech", "Stock Market", "Consumer Market",
                           "Economy", "Investing", "Personal Finance")

lda.terms
```

```
##       Public Policy Politics  Consumer Finance/Debt Energy    Tech
##  [1,] "rate"        "loan"    "obama"               "share"   "busi"
##  [2,] "price"       "water"   "presid"              "billion" "servic"
##  [3,] "oil"         "credit"  "republican"          "quarter" "peopl"
##  [4,] "fed"         "vehicl"  "state"               "earn"    "million"
##  [5,] "market"      "student" "hous"                "revenu"  "product"
##  [6,] "growth"      "car"     "tax"                 "million" "game"
##  [7,] "increas"     "million" "senat"               "expect"  "googl"
##  [8,] "economi"     "busi"    "govern"              "analyst" "work"
##  [9,] "expect"      "secur"   "democrat"            "sale"    "market"
## [10,] "job"         "consum"  "congress"            "apple"   "content"
##       Stock Market Consumer Market Economy    Investing  Personal Finance
##  [1,] "home"       "peopl"         "drug"     "state"    "fund"
##  [2,] "sale"       "retir"         "peopl"    "tax"      "bank"
##  [3,] "retail"     "insur"         "work"     "ticket"   "market"
##  [4,] "price"      "financi"       "vaccin"   "airlin"   "stock"
##  [5,] "store"      "health"        "develop"  "law"      "investor"
##  [6,] "hous"       "save"          "patient"  "york"     "invest"
##  [7,] "citi"       "cost"          "research" "investig" "manag"
##  [8,] "market"     "pay"           "user"     "court"    "bond"
##  [9,] "mortgag"    "tax"           "health"   "million"  "capit"
## [10,] "weather"    "money"         "show"     "flight"   "billion"
```

2. Retrieve new articles from CNBC homepage

```
news = read_html("http://www.cnbc.com/us-news/")
foo = html_nodes(news, ".headline a")
foo = html_attr(foo, "href")
foo = foo[!is.na(foo)]
foo = foo[grep("^/", foo)]
```

Retrieve text from each article

```
store_text = NULL

for(i in 1:length(foo)){
  text = read_html(paste("http://www.cnbc.com", foo[i], sep=""))
  text = html_nodes(text, "p")
  text = html_text(text)
  text = text[-1]
  text = paste(text, sep="", collapse="")
  store_text[i] = text
}
```

Creating a new corpus with the articles, cleaning up the corpus

```
corp.foo <- VCorpus(VectorSource(store_text))
corp.foo <- tm_map(corp.foo, removePunctuation)
corp.foo <- tm_map(corp.foo, removeNumbers)
corp.foo <- tm_map(corp.foo, content_transformer(removeWords), c("TIL")  ,lazy=TRUE)
corp.foo <- tm_map(corp.foo, content_transformer(removeWords), stopwords("SMART"),lazy=TRUE)
corp.foo <- tm_map(corp.foo, content_transformer(tolower) ,lazy=TRUE)
corp.foo <- tm_map(corp.foo, content_transformer(stemDocument) ,lazy=TRUE)
myStopwords <- c("percent", "plan", "year", "read", "compani", "inform", "time", "day", "make", "the",
corp.foo <- tm_map(corp.foo, removeWords, myStopwords)
corp.foo <- tm_map(corp.foo, stripWhitespace)
```

3. Classify news articles using your topic model

Creating a new document term matrix, using only words appearing on the original DTM

```
dic = Terms(dtm)
new_dtm = DocumentTermMatrix(corp.foo, control=list(dictionary = dic))
new_dtm = new_dtm[rowSums(as.matrix(new_dtm))!=0,]
```

Probabilities of each article belonging to a topic LDA topic

```
topic_probabilities = posterior(ldaOut, new_dtm)
topic_probabilities$topics
```

```
##               1            2            3            4            5
## 1  0.0490169196 0.0003759458 0.0003759458 0.0003759458 0.2690132119
## 2  0.0836818487 0.0976510150 0.1627687280 0.0240446208 0.0727171443
## 3  0.0001967429 0.0424852239 0.0001967429 0.0201942901 0.2614989722
## 4  0.0003211060 0.0003211060 0.2456792313 0.3694603424 0.1419832787
## 5  0.0004197548 0.0004197548 0.3579264412 0.0004197548 0.0004197548
## 6  0.6628917348 0.0003291072 0.1082015485 0.0946517633 0.0003291072
## 7  0.0003621996 0.0924205794 0.5323267986 0.0003621996 0.0003621996
## 8  0.0003494232 0.0003494232 0.2697183620 0.0003494232 0.0003494232
## 9  0.0687171086 0.0006058953 0.1378991875 0.5095965085 0.2801518233
## 10 0.0004990573 0.2025083295 0.0004990573 0.0004990573 0.3706615875
## 11 0.0001492986 0.0001492986 0.5406530307 0.0313473635 0.1182951610
## 12 0.0288486578 0.0001350504 0.4656169189 0.0822965279 0.1625850510
## 13 0.0002381260 0.1746810287 0.0668998085 0.0002381260 0.4791740173
## 14 0.2179239871 0.0003086015 0.0003086015 0.0726414078 0.1102930711
## 15 0.0157763857 0.0767541730 0.0001545392 0.0319391523 0.4190899324
## 16 0.0002782607 0.0483341612 0.4053302944 0.0002782607 0.1205343600
## 17 0.0010595525 0.2799688289 0.2296638799 0.3751617497 0.0010595525
## 18 0.0796138168 0.0001948083 0.0661929892 0.4168633863 0.0673861341
## 19 0.0292329595 0.0233634846 0.4883661136 0.0001310310 0.0150500884
## 20 0.0008360005 0.1040991944 0.0008360005 0.0008360005 0.4629590940
## 21 0.0002272026 0.0002272026 0.6346472620 0.0002272026 0.0002272026
## 22 0.0020428334 0.0020428334 0.7439530939 0.0020428334 0.0020428334
## 23 0.6800365469 0.0001967429 0.0802533141 0.0429800889 0.0839474525
##               6            7            8            9           10
## 1  0.0003759458 0.0688035568 0.4041460974 0.2071404852 0.0003759458
## 2  0.0858538229 0.2687195841 0.0001009777 0.1684390310 0.0360232274
## 3  0.2493992104 0.3594533053 0.0661820267 0.0001967429 0.0001967429
## 4  0.0003211060 0.0003211060 0.0003211060 0.0641879167 0.1770837008
## 5  0.0004197548 0.5868158587 0.0004197548 0.0523194165 0.0004197548
## 6  0.0003291072 0.0299082769 0.0003291072 0.0003291072 0.1027011403
## 7  0.0652274478 0.0552905701 0.0003621996 0.1820507957 0.0712350100
## 8  0.0003494232 0.1365287029 0.1092070978 0.4437779439 0.0390207775
## 9  0.0006058953 0.0006058953 0.0006058953 0.0006058953 0.0006058953
## 10 0.1358655196 0.0485248147 0.2206523486 0.0197911707 0.0004990573
## 11 0.0783076112 0.0001492986 0.0437186548 0.0700992831 0.1171310001
## 12 0.0477773488 0.0001350504 0.0913332687 0.0959574178 0.0253147083
## 13 0.0305703131 0.0002381260 0.2474842022 0.0002381260 0.0002381260
## 14 0.0588233789 0.0822302177 0.0003086015 0.0380635844 0.4190985485
## 15 0.0001545392 0.0594910272 0.2777398949 0.0767156103 0.0421847457
## 16 0.0002782607 0.0648688790 0.1240057067 0.2358135560 0.0002782607
## 17 0.0010595525 0.0010595525 0.1088482262 0.0010595525 0.0010595525
## 18 0.0001948083 0.1313387760 0.0001948083 0.0001948083 0.2378256644
## 19 0.0001310310 0.0328782623 0.1067597621 0.2771091863 0.0269780812
## 20 0.4270897083 0.0008360005 0.0008360005 0.0008360005 0.0008360005
## 21 0.0002272026 0.2364501837 0.0002272026 0.1273121363 0.0002272026
## 22 0.0020428334 0.2397042385 0.0020428334 0.0020428334 0.0020428334
## 23 0.0001967429 0.0001967429 0.0001967429 0.0156476358 0.0963479904
```

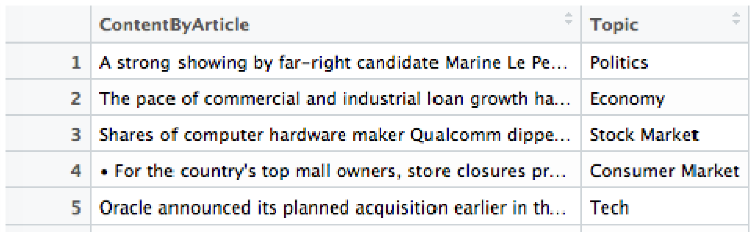
For each topic, we select the highest probability of belonging to a topic

```r
topic_by_article = colnames(topic_probabilities$topics)[max.col(topic_probabilities$topics,
                                             ties.method="first")]
topic_by_article ##outputs the topic number for each article
```

```
##  [1] "8"  "7"  "7"  "4"  "7"  "1"  "3"  "9"  "4"  "5"  "3"  "3"  "5"  "10"
## [15] "5"  "3"  "4"  "4"  "3"  "5"  "3"  "3"  "1"
```

Creating a table, with 10 articles, and their topic model classification

```r
final_table = NULL
final_table$ContentByArticle = store_text[1:10]
final_table$Topic = colnames(lda.terms)[as.numeric(topic_by_article[1:10])]
```



	ContentByArticle	Topic
1	A strong showing by far-right candidate Marine Le Pe...	Politics
2	The pace of commercial and industrial loan growth ha...	Economy
3	Shares of computer hardware maker Qualcomm dippe...	Stock Market
4	• For the country's top mall owners, store closures pr...	Consumer Market
5	Oracle announced its planned acquisition earlier in th...	Tech