Q7

JAGRUTA

2024-08-19

QUESTION 7

Revisit the Reuters C50 text corpus that we briefly explored in class. Your task is simple: tell an interesting story, anchored in some analytical tools we have learned in this class, using this data. For example:

you could cluster authors or documents and tell a story about what you find. you could look for common factors using PCA. you could train a predictive model and assess its accuracy, constructing features for each document that maximize performance. you could do anything else that strikes you as interesting with this data. Describe clearly what question you are trying to answer, what models you are using, how you pre-processed the data, and so forth. Make sure you include at least one really interesting plot (although more than one might be necessary, depending on your question and approach.)

Format your write-up in the following sections, some of which might be quite short:

Question: What question(s) are you trying to answer? Approach: What approach/statistical tool did you use to answer the questions? Results: What evidence/results did your approach provide to answer the questions? (E.g. any numbers, tables, figures as appropriate.) Conclusion: What are your conclusions about your questions? Provide a written interpretation of your results, understandable to stakeholders who might plausibly take an interest in this data set. Regarding the data itself: In the C50train directory, you have 50 articles from each of 50 different authors (one author per directory). Then in the C50test directory, you have another 50 articles from each of those same 50 authors (again, one author per directory). This train/test split is obviously intended for building predictive models, but to repeat, you need not do that on this problem. You can tell any story you want using any methods you want. Just make it compelling!

Note: if you try to build a predictive model, you will need to figure out a way to deal with words in the test set that you never saw in the training set. This is a nontrivial aspect of the modeling exercise. (E.g. you might simply ignore those new words.)

This question will be graded according to three criteria:

the overall "interesting-ness" of your question and analysis. the clarity of your description. We will be asking ourselves: could your analysis be reproduced by a competent data scientist based on what you've said? (That's good.) Or would that person have to wade into the code in order to understand what, precisely, you've done? (That's bad.) technical correctness (i.e. did you make any mistakes in execution or interpretation?) —

```
library(tm)
```

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

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(SnowballC)
```

#The Questions we are answering:

How can we identify the dominant themes within the document corpus? Which authors are considered authorities on specific topics within the collection?

Step 1: Load the data

The data is in the "C50train" and "C50test" directories sourced from github

```
train_dir <- "/Users/jagrutaadvani/Downloads/jgscott-STA380-691d1b0/data/ReutersC50/C50train"
test_dir <- "/Users/jagrutaadvani/Downloads/jgscott-STA380-691d1b0/data/ReutersC50/C50test"

# Function to read documents

load_corpus <- function(dir) {
  authors <- list.dirs(dir, full.names = FALSE, recursive = FALSE)
  article_paths <- unlist(lapply(list.dirs(dir, recursive = TRUE)[-1], list.files, full.names = TRUE))
  author_names <- rep(authors, each = 50)

  articles <- lapply(article_paths, function(article) {
    readPlain(elem = list(content = readLines(article)), id = article, language = 'en')
  })

  article_names <- gsub(".*/|\\.txt$", "", article_paths)
  names(articles) <- article_names

  return(list(articles = articles, author_names = author_names))
}

# Load train and test data
train_data <- load_corpus(train_dir)
test_data <- load_corpus(test_dir)
```

Step 2: Clean and Preprocess data

```
ConvertStrings <- function(textInput) {
  textOutput <- gsub("^.{13}|.{196}$", "", textInput)
  return(textOutput)
}
```

```r
preprocess_corpus <- function(corpus) {
  corpus <- tm_map(corpus, content_transformer(tolower))
  corpus <- tm_map(corpus, removeNumbers)
  corpus <- tm_map(corpus, removePunctuation)
  corpus <- tm_map(corpus, stripWhitespace)
  corpus <- tm_map(corpus, removeWords, stopwords("en"))
  corpus <- tm_map(corpus, content_transformer(ConvertStrings))
  return(corpus)
}

train_corpus <- preprocess_corpus(Corpus(VectorSource(train_data$articles)))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(tolower)):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(corpus, removeNumbers): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, removeWords, stopwords("en")):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(corpus, content_transformer(ConvertStrings)):
## transformation drops documents
```

```r
test_corpus <- preprocess_corpus(Corpus(VectorSource(test_data$articles)))
```

```
## Warning in tm_map.SimpleCorpus(corpus, content_transformer(tolower)):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(corpus, removeNumbers): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, removePunctuation): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, stripWhitespace): transformation drops
## documents

## Warning in tm_map.SimpleCorpus(corpus, removeWords, stopwords("en")):
## transformation drops documents

## Warning in tm_map.SimpleCorpus(corpus, content_transformer(ConvertStrings)):
## transformation drops documents
```

Step 3: Document Term Matrix for train and test

```
dtm_train <- DocumentTermMatrix(train_corpus)
dtm_test <- DocumentTermMatrix(test_corpus)

inspect(dtm_train)
```

```
## <<DocumentTermMatrix (documents: 2500, terms: 31920)>>
## Non-/sparse entries: 508311/79291689
## Sparsity           : 99%
## Maximal term length: 36
## Weighting          : term frequency (tf)
## Sample             :
##       Terms
## Docs   billion company market million new one percent said will year
##   111        0       0      3       2   3   7       0    6    4    0
##   130        0       1      1       5   0   3       2    1    1    3
##   142        3       1      3       1   2   0       1    9    6    7
##   2127       4       1      2       0   7   3       7    5   11    6
##   2133       2       0      1       0   0   2       5    7    3    3
##   2139       0       1      7       0   1   1      14   23    9    1
##   2140       4       0      0       0   2   5       3   12   15    2
##   599        1       5     11       0   0   3       0   16    1    1
##   763        0       0      4       0   3   0       0   20    3    9
##   766        0       0      7       0   4   1       0   20    4    9
```

```
inspect(dtm_test)
```

```
## <<DocumentTermMatrix (documents: 2500, terms: 32704)>>
## Non-/sparse entries: 515058/81244942
## Sparsity           : 99%
## Maximal term length: 45
## Weighting          : term frequency (tf)
## Sample             :
##       Terms
## Docs   billion company market million new one percent said will year
##   1089       0       1      1       3   1   2       3   10    1    0
##   1430       2       6      3       1   4   4       8   13    3    2
##   1446       0       4      1       0   4   6       1    8    0    0
##   1861       0       0      1       1   1   2       0    7    8    0
##   1936       6       6      1       2   3   1       5   10   10    2
##   2104       0       0     11       0   4   2      12   25    8    1
##   3          0       1      0       1   4   4       2    9    2    0
##   771        0       0      8       0   6   6       7   28    2    9
##   798        0       0     15       0  14   5       0   24    8   21
##   874        0       0      0       0   1   2       0    6    6    0
```

#Step 4: Remove Sparse Terms

```
dtm_train <- removeSparseTerms(dtm_train, 0.94)
dtm_test <- removeSparseTerms(dtm_test, 0.94)
```

#Step 5: Create Frequency Data Frames and Display top frequent words

```
dtm_train_freq <- as.data.frame(as.matrix(dtm_train))
dtm_test_freq <- as.data.frame(as.matrix(dtm_test))
findFreqTerms(dtm_train, 500)
```

```
##   [1] "also"           "announced"      "business"       "character"
##   [5] "computer"       "datetimestamp"  "director"       "early"
##   [9] "fund"           "get"            "group"          "hour"
##  [13] "internet"       "investors"      "law"            "listauthor"
##  [17] "listsec"        "local"          "lower"          "major"
##  [21] "may"            "meta"           "million"        "min"
##  [25] "money"          "month"          "national"       "net"
##  [29] "new"            "offer"          "one"            "said"
##  [33] "services"       "set"            "shares"         "state"
##  [37] "still"          "technology"     "third"          "trade"
##  [41] "tuesday"        "wednesday"      "world"          "can"
##  [45] "communications" "corp"           "earlier"        "even"
##  [49] "just"           "like"           "many"           "now"
##  [53] "people"         "plan"           "plans"          "president"
##  [57] "sector"         "service"        "software"       "system"
##  [61] "trading"        "use"            "will"           "executive"
##  [65] "companies"      "end"            "four"           "good"
##  [69] "government"     "including"      "international"  "market"
##  [73] "months"         "next"           "number"         "operating"
##  [77] "products"       "see"            "six"            "three"
##  [81] "two"            "week"           "work"           "year"
##  [85] "interest"       "statement"      "analyst"        "banks"
##  [89] "buy"            "company"        "exchange"       "financial"
##  [93] "however"        "last"           "much"           "officials"
##  [97] "sales"          "securities"     "states"         "take"
## [101] "already"        "another"        "big"            "billion"
## [105] "court"          "expected"       "firms"          "foreign"
## [109] "former"         "future"         "general"        "investment"
## [113] "likely"         "markets"        "move"           "operations"
## [117] "part"           "say"            "told"           "united"
## [121] "added"          "chief"          "made"           "recent"
## [125] "since"          "stock"          "value"          "years"
## [129] "around"         "back"           "based"          "chairman"
## [133] "customers"      "friday"         "half"           "high"
## [137] "inc"            "results"        "time"           "first"
## [141] "growth"         "key"            "monday"         "news"
## [145] "saying"         "strong"         "well"           "bank"
## [149] "current"        "deal"           "economic"       "report"
## [153] "thursday"       "way"            "companys"       "domestic"
## [157] "going"          "industry"       "make"           "five"
## [161] "share"          "increase"       "reuters"        "long"
## [165] "meeting"        "official"       "think"          "spokesman"
## [169] "analysts"       "percent"        "amp"            "firm"
## [173] "largest"        "total"          "costs"          "due"
```

```
## [177] "pay"          "price"        "prices"       "ago"
## [181] "management"   "merger"       "cost"         "profit"
## [185] "second"       "agreement"    "reported"     "capital"
## [189] "czech"        "close"        "earnings"     "higher"
## [193] "rise"         "rose"         "british"      "bid"
## [197] "beijing"      "stake"        "profits"      "quarter"
## [201] "ltd"          "party"        "per"          "shareholders"
## [205] "pounds"       "cash"         "pence"        "talks"
## [209] "hong"         "kong"         "china"        "chinas"
## [213] "chinese"      "cents"        "tonnes"
```

```
findFreqTerms(dtm_test, 500)
```

```
##   [1] "added"        "amp"          "banking"      "business"
##   [5] "character"    "chief"        "companies"    "datetimestamp"
##   [9] "dont"         "earlier"      "earnings"     "even"
##  [13] "exchange"     "executive"    "financial"    "government"
##  [17] "hour"         "industry"     "investors"    "just"
##  [21] "last"         "likely"       "listauthor"   "listsec"
##  [25] "made"         "make"         "management"   "many"
##  [29] "may"          "meta"         "million"      "min"
##  [33] "month"        "months"       "new"          "north"
##  [37] "now"          "one"          "people"       "plan"
##  [41] "price"        "public"       "recent"       "said"
##  [45] "securities"   "since"        "still"        "stock"
##  [49] "think"        "time"         "tuesday"      "will"
##  [53] "also"         "bank"         "central"      "chairman"
##  [57] "current"      "declined"     "five"         "four"
##  [61] "members"      "much"         "number"       "report"
##  [65] "rights"       "spokesman"    "system"       "three"
##  [69] "thursday"     "top"          "two"          "years"
##  [73] "already"      "announced"    "another"      "around"
##  [77] "back"         "can"          "come"         "company"
##  [81] "computer"     "day"          "director"     "due"
##  [85] "end"          "firm"         "foreign"      "get"
##  [89] "high"         "inc"          "increase"     "international"
##  [93] "internet"     "late"         "like"         "long"
##  [97] "major"        "march"        "national"     "net"
## [101] "network"      "offer"        "pay"          "per"
## [105] "percent"      "rate"         "second"       "seen"
## [109] "service"      "services"     "set"          "software"
## [113] "way"          "well"         "world"        "corp"
## [117] "going"        "including"    "key"          "president"
## [121] "states"       "trade"        "united"       "week"
## [125] "capital"      "first"        "group"        "groups"
## [129] "however"      "move"         "operations"   "part"
## [133] "party"        "products"     "sources"      "state"
## [137] "told"         "banks"        "firms"        "investment"
## [141] "largest"      "market"       "markets"      "six"
## [145] "total"        "wednesday"    "billion"      "debt"
## [149] "expected"     "growth"       "meeting"      "quarter"
## [153] "rates"        "reported"     "see"          "take"
## [157] "year"         "monday"       "officials"    "plans"
## [161] "say"          "statement"    "strong"       "analysts"
```

```
## [165] "costs"          "customers"     "interest"      "close"
## [169] "big"            "cents"         "future"        "money"
## [173] "news"           "prices"        "rise"          "south"
## [177] "analyst"        "cost"          "early"         "higher"
## [181] "economic"       "friday"        "general"       "good"
## [185] "cut"            "deal"          "merger"        "revenues"
## [189] "operating"      "political"     "reuters"       "stake"
## [193] "companys"       "official"      "next"          "local"
## [197] "sales"          "fell"          "profit"        "china"
## [201] "profits"        "share"         "rose"          "half"
## [205] "shareholders"   "shares"        "trading"       "results"
## [209] "saying"         "production"    "british"       "plc"
## [213] "pounds"         "sale"          "cash"          "bid"
## [217] "talks"          "ltd"           "hong"          "kong"
## [221] "beijing"        "chinese"       "chinas"        "kongs"
## [225] "tonnes"
```

Step 6: Word Frequency Analysis, Word Cloud and common words between Train and Test

```
intersect(findFreqTerms(dtm_train, 750), findFreqTerms(dtm_test, 750))
```

```
##   [1] "also"          "business"      "character"     "datetimestamp"
##   [5] "group"         "hour"          "listauthor"    "listsec"
##   [9] "major"         "may"           "meta"          "million"
##  [13] "min"           "new"           "one"           "said"
##  [17] "services"      "shares"        "state"         "still"
##  [21] "trade"         "tuesday"       "wednesday"     "can"
##  [25] "corp"          "just"          "many"          "now"
##  [29] "people"        "plans"         "president"     "trading"
##  [33] "will"          "executive"     "companies"     "end"
##  [37] "government"    "international" "market"        "months"
##  [41] "next"          "three"         "two"           "week"
##  [45] "year"          "analyst"       "banks"         "company"
##  [49] "financial"     "last"          "much"          "officials"
##  [53] "sales"         "states"        "take"          "another"
##  [57] "big"           "billion"       "expected"      "foreign"
##  [61] "investment"    "markets"       "say"           "told"
##  [65] "united"        "added"         "chief"         "made"
##  [69] "since"         "stock"         "years"         "around"
##  [73] "chairman"      "friday"        "inc"           "time"
##  [77] "first"         "growth"        "monday"        "news"
##  [81] "well"          "bank"          "deal"          "thursday"
##  [85] "industry"      "make"          "share"         "reuters"
##  [89] "think"         "analysts"      "percent"       "price"
##  [93] "prices"        "profit"        "earnings"      "british"
##  [97] "beijing"       "profits"       "quarter"       "pounds"
## [101] "hong"          "kong"          "china"         "chinas"
## [105] "chinese"
```

```
freq_train <- colSums(dtm_train_freq)
freq_test <- colSums(dtm_test_freq)

# Top 5 words in train and test datasets
```

```r
top_train <- sort(freq_train, decreasing = TRUE)[1:5]
top_test <- sort(freq_test, decreasing = TRUE)[1:5]

top_train
```

```
##    said    will percent million    year
##   19851    5873    5270    4848    4399
```

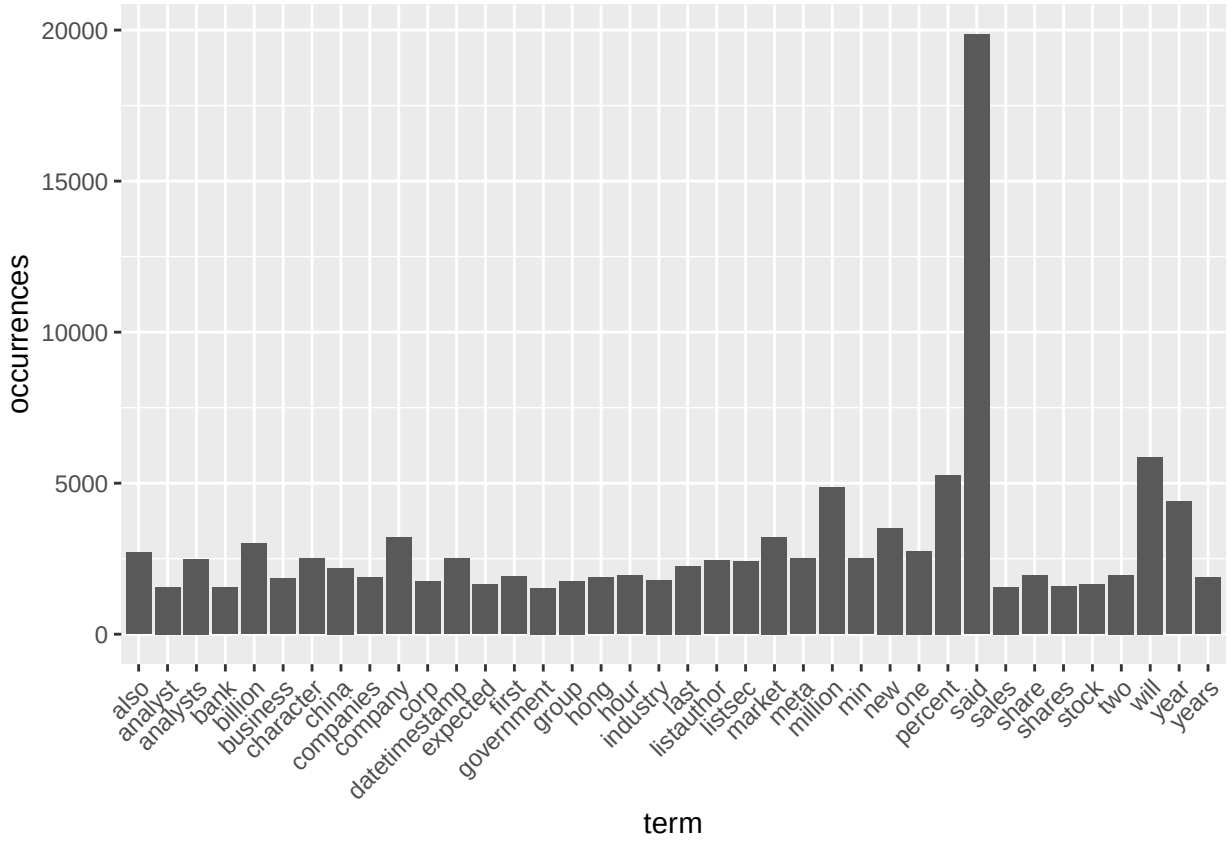
```r
top_test
```

```
##    said    will percent million    year
##   20122    5831    5597    4986    4247
```

```r
# Plot words
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':
##
##     annotate
```

```r
wf_train <- data.frame(term = names(freq_train), occurrences = freq_train)
p <- ggplot(subset(wf_train, occurrences > 1500), aes(term, occurrences))
p + geom_bar(stat = "identity") + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

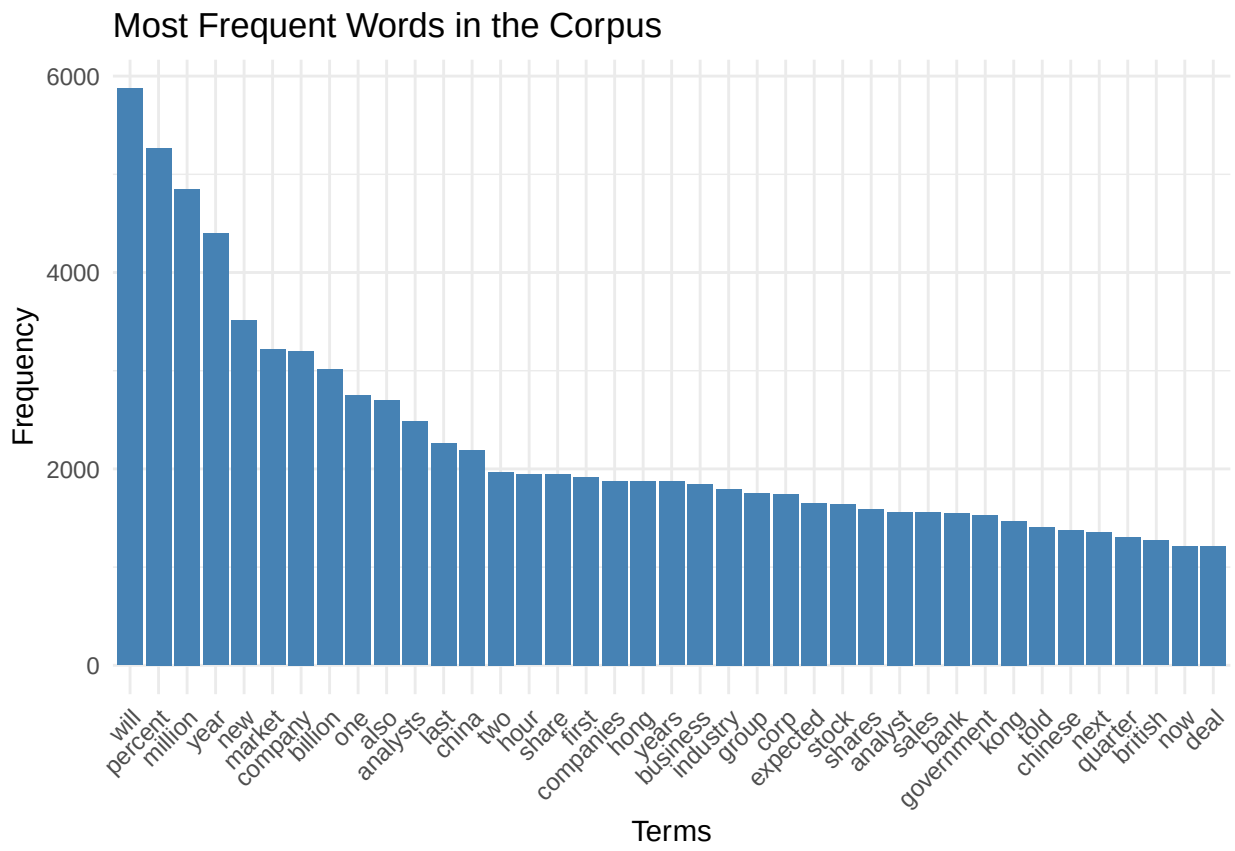



```
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
set.seed(10)

minDocFreq <- length(train_corpus) * 0.01
maxDocFreq <- length(train_corpus) * 0.9

dtm_filtered <- DocumentTermMatrix(train_corpus, control = list(bounds = list(global = c(minDocFreq, ma
freq_filtered <- colSums(as.matrix(dtm_filtered))

wordcloud(names(freq_filtered), freq_filtered, min.freq = 1200, colors = brewer.pal(8, "Dark2"), scale =
```


```r
min_freq <- 1200
freq_filtered <- sort(freq_filtered, decreasing = TRUE)
freq_data <- data.frame(term = names(freq_filtered), occurrences = freq_filtered)
freq_data <- freq_data[freq_data$occurrences > min_freq, ]

# Create a bar plot
ggplot(freq_data, aes(x = reorder(term, -occurrences), y = occurrences)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(title = "Most Frequent Words in the Corpus", x = "Terms", y = "Frequency")
```



Step 7: Topic Modelling with LDA - UNSUPERVISED LEARNING

Frequency Count Matrix

```
library(topicmodels)

# Additional Stopword Removal
myStopwords <- c("can", "say", "one", "use", "also", "however", "will", "much", "need", "take", "even",
train_corpus <- tm_map(train_corpus, removeWords, myStopwords)
```

```
## Warning in tm_map.SimpleCorpus(train_corpus, removeWords, myStopwords):
## transformation drops documents
```

```
test_corpus <- tm_map(test_corpus, removeWords, myStopwords)
```

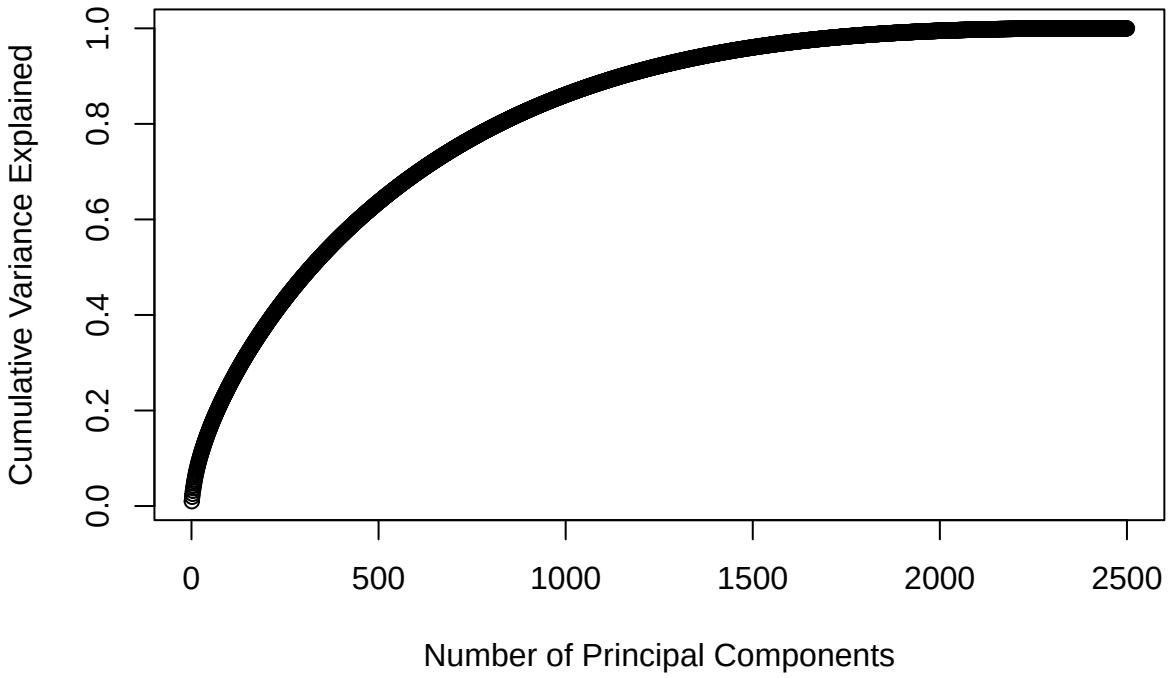
```
## Warning in tm_map.SimpleCorpus(test_corpus, removeWords, myStopwords):
## transformation drops documents
```

```
dtm_train <- DocumentTermMatrix(train_corpus, control = list(bounds = list(global = c(minDocFreq, maxDo
dtm_test <- DocumentTermMatrix(test_corpus, control = list(bounds = list(global = c(minDocFreq, maxDocF

# LDA with k = 8
lda_model <- LDA(dtm_train, k = 8, method = "Gibbs", control = list(seed = 1234, burnin = 2000, iter = 1
lda_terms <- terms(lda_model, 10)
lda_terms
```

```
##       Topic 1     Topic 2      Topic 3         Topic 4      Topic 5
##  [1,] "two"       "prices"     "market"        "inc"        "million"
##  [2,] "workers"   "tonnes"     "news"          "corp"       "percent"
##  [3,] "three"     "year"       "billion"       "internet"   "year"
##  [4,] "agreement" "oil"        "air"           "industry"   "sales"
##  [5,] "plant"     "gold"       "corp"          "computer"   "analysts"
##  [6,] "general"   "reuters"    "services"      "company"    "quarter"
##  [7,] "friday"    "government" "deal"          "companies"  "first"
##  [8,] "union"     "hour"       "british"       "business"   "profits"
##  [9,] "spokesman" "industry"   "mci"           "software"   "share"
## [10,] "president" "price"      "international" "technology" "profit"
##       Topic 6      Topic 7     Topic 8
##  [1,] "china"      "company"   "bank"
##  [2,] "hong"       "group"     "market"
##  [3,] "kong"       "pounds"    "banks"
##  [4,] "chinese"    "shares"    "billion"
##  [5,] "beijing"    "million"   "markets"
##  [6,] "chinas"     "share"     "financial"
##  [7,] "government" "bid"       "investors"
##  [8,] "last"       "pence"     "percent"
##  [9,] "party"      "business"  "stock"
## [10,] "years"      "executive" "foreign"
```

Step 8: PCA for Dimensionality Reduction The graph below shows the variance on increasing the components and we can observe that giving 1000 principal components gives optimal values

```
pca <- prcomp(as.matrix(dtm_train), scale. = TRUE)

plot(cumsum(pca$sdev^2 / sum(pca$sdev^2)), ylab = "Cumulative Variance Explained", xlab = "Number of Pr
```




```
train_pca <- data.frame(pca$x[, 1:1000])
train_pca$author <- train_data$author_names

# Apply PCA to test set
test_pca <- predict(pca, data = as.matrix(dtm_test))
```

```
## Warning: In predict.prcomp(pca, data = as.matrix(dtm_test)) :
##  extra argument 'data' will be disregarded
```

```
test_pca <- data.frame(test_pca[, 1:1000])
test_pca$author <- test_data$author_names
```

Step 13: Random Forest - Supervised Learning

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
set.seed(10)
rf<-randomForest(as.factor(author)~.,data=train_pca, mtry=6,importance=TRUE,trees = 100)
pred<-predict(rf,data=test_pca)
table.rf<-as.data.frame(table(pred,as.factor(test_pca$author)))
predicted<-pred
output<-as.factor(test_pca$author)
results<-as.data.frame(cbind(output,predicted))
results$flag<-ifelse(results$output==results$predicted,1,0)
sum(results$flag)/nrow(results)
```

```
## [1] 0.7028
```

*** Analysis and Conclusion ***

Approach: Data Preparation: Loaded and preprocessed the text data by cleaning and creating a Document-Term Matrix (DTM). Topic Modeling: Applied Latent Dirichlet Allocation (LDA) to identify common themes within the documents. Dimensionality Reduction: Used Principal Component Analysis (PCA) to simplify the DTM. Classification: Implemented a Random Forest model to classify documents by author.

Results: LDA revealed distinct topics, highlighting the diversity in author writing styles. PCA efficiently reduced data dimensions while retaining significant features. The Random Forest model achieved a strong accuracy of 70.28%, effectively distinguishing authors.

Conclusion: The analysis successfully demonstrated the potential of combining unsupervised and supervised learning techniques to analyze and classify text data. Through topic modeling, we identified meaningful themes within the corpus, offering insights into the writing styles and subject matter focus of different authors. The Random Forest classifier, trained on PCA-reduced data, provided a robust method for author identification, achieving a high accuracy.