```
---
title: "reuters_again"
author: "Lowrance, Mikala"
date: "2024-08-18"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

R Markdown


```{r Reuters}
library(tm)
library(tidyverse)

readerPlain = function(fname){
  readPlain(elem=list(content=readLines(fname)),
            id=fname, language='en') }

file_list = Sys.glob('repos/STA380/data/ReutersC50/C50train/*/*.txt')
reuters = lapply(file_list, readerPlain)

# Clean up the file names
mynames = file_list %>%
  { strsplit(., '/', fixed=TRUE) } %>%
  { lapply(., tail, n=2) } %>%
  { lapply(., paste0, collapse = '') } %>%
  unlist


# Extract author names from file paths
author_names = file_list %>%
  { strsplit(., '/', fixed=TRUE) } %>%
  { lapply(., function(x) x[length(x) - 1]) } %>%
  unlist

# Rename the articles
names(reuters) = mynames

# Create corpus
documents_raw = Corpus(VectorSource(reuters))

# Pre-processing steps
my_documents = documents_raw
my_documents = tm_map(my_documents, content_transformer(tolower))
my_documents = tm_map(my_documents, content_transformer(removeNumbers))
my_documents = tm_map(my_documents, content_transformer(removePunctuation))
my_documents = tm_map(my_documents, content_transformer(stripWhitespace))


# Remove stop words
#stopwords("en")
#stopwords("SMART")
my_documents = tm_map(my_documents, content_transformer(removeWords), stopwords("en"))
my_documents = tm_map(my_documents, content_transformer(removeWords), stopwords("SMART"))


# Create document-term matrix
DTM_reuters = DocumentTermMatrix(my_documents)
```
```

```
class(DTM_reuters)

# Review frequent words & word associations
#inspect(DTM_reuters[1:10,1:20])
findFreqTerms(DTM_reuters, lowfreq = 500)
findAssocs(DTM_reuters, "approve", .5)

# Remove infrequent terms
DTM_reuters = removeSparseTerms(DTM_reuters, 0.95)
DTM_reuters

# Create TF-IDF weights
tfidf_reuters = weightTfIdf(DTM_reuters)

# Compare documents
#inspect(tfidf_reuters[1,])

####
# Dimensionality reduction
####

# PCA on term frequencies
X = as.matrix(tfidf_reuters)
summary(colSums(X))
scrub_cols = which(colSums(X) == 0)
X = X[,-scrub_cols]

pca_reuters = prcomp(X, rank=2, scale=TRUE)
plot(pca_reuters)

# Look at the loadings
pca_reuters$rotation[order(abs(pca_reuters$rotation[,1]),decreasing=TRUE),1][1:25]
pca_reuters$rotation[order(abs(pca_reuters$rotation[,2]),decreasing=TRUE),2][1:25]

# Look at the first two PCs
#pca_reuters$x[,1:2]

plot(pca_reuters$x[,1:2], xlab="PCA 1 direction", ylab="PCA 2 direction", bty="n",
     type='n')
text(pca_reuters$x[,1:2], labels = 1:length(reuters), cex=0.7)

# Cluster documents

# define the distance matrix
# using the PCA scores
dist_mat = dist(pca_reuters$x)
tree_reuters = hclust(dist_mat)
#plot(tree_reuters)
clust5 = cutree(tree_reuters, k=5)

# Inspect the clusters
which(clust5 == 5)
content(reuters[[651]])
content(reuters[[652]])

##########
# Create a data frame for plotting
df <- data.frame(
  PC1 = pca_reuters$x[,1],  # First principal component
  PC2 = pca_reuters$x[,2],  # Second principal component
  Author = as.factor(author_names)  # Author names as factors
)

# Try clustering with authors
```

```
df <- data.frame(
  PC1 = pca_reuters$x[,1],  # First principal component
  PC2 = pca_reuters$x[,2],  # Second principal component
  Author = as.factor(author_names)  # Author names as factors
)

# Subset the data by author
authors <- unique(author_names)
clusters_by_author <- list()

for (author in authors) {
  # Subset documents by author
  subset_df <- df[df$Author == author, ]

  # Perform PCA and clustering on this subset
  subset_dist <- dist(subset_df[, 1:2])
  subset_hclust <- hclust(subset_dist)

  # Cut into clusters
  clusters_by_author[[author]] <- cutree(subset_hclust, k = 5)  # Adjust k as needed
}

library(ggplot2)
df$Cluster <- as.factor(clust5)

ggplot(df, aes(x = PC1, y = PC2, color = Cluster, shape = Author)) +
  geom_point(size = 3) +
  labs(title = "PCA of Reuters Documents", x = "PCA 1", y = "PCA 2") +
  theme_minimal()

# Create a table of cluster assignments for each author
table(df$Author, df$Cluster)

```
Question: What reuters author write about similar topics? If I like the writing of a specific writer in Reuters, what other authors should I read?

Approach: We created a corpus of reuters documents from 42 authors. First, we used a tokenization approach on every word and compiled the words into a document term matrix. After that, we removed all of the infrequent words and commonly used "stop words". Lastly, we created TF-IDF weights for the terms in the DTM to appropriately weigh frequent words within a specific document, yet rare across the corpus. After completing these pre-processing steps, we applied principle component analysis (PCA) to the reduce dimensionality and allow a visualization of the distribution of documents within a 2D space. We kept only 2 principle components to easily view relationships and clusters. To finally view the similarity of authors, we clustered the documents into 5 categories based on the PCA results.

Conclusion: This graph allows us to see the similarity of what authors write about, and where there is crossover. For example, Benjamin Kang Lim is the only writer in cluster 5, while he has a few documents that fall into cluster 3 and are similar to the majority of Alan Crosby's work. Conversely to cluster 5, cluster 4 has several authors with similar documents. Reuters and other publishing companies could use this analysis to recommend similar authors to a reader.