Lab Section 6: Vector Space & Clustering in R

Napon Jatusripitak

5/10/2019

Distance/Similarity

ASEAN Member States

Recall the ASEAN Member States dataframe that you compiled from last week.

Suppose that we are interested in measuring the distance or similarity between the wiki pages for each countries in ASEAN.

Step 1: Importing the Data

```
setwd("~/Documents/GitHub/MMSS_311_2/Lab Exercises/Week 6")
df <- read.csv("asean.csv", stringsAsFactors = F)
```

Step 2: Preprocessing, Tokenizing and Converting to DTM (tm)

First, we have to prepare the data for use with `tm`. Since we'll be using the function `VCorpus(DataframeSource())`, it is necessary to make adjustments to our dataframe to ensure that doc_id is in the first column and text is in the second column.

```r
df <- df %>%
  mutate(doc_id = row_number()) %>%
  select(doc_id, text, everything())
```

Step 2: Preprocessing, Tokenizing and Converting to DTM (tm)

Next, we convert the dataframe to a corpus object. Preprocess accordingly.

```r
corpus <- VCorpus(DataframeSource(df)) %>%
  tm_map(removePunctuation) %>%
  tm_map(removeNumbers) %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(removeWords, stopwords('english')) %>%
  tm_map(stemDocument) %>%
  tm_map(stripWhitespace)
```

Step 2: Preprocessing, Tokenizing and Converting to DTM (tm)

Lastly, we convert the corpus to a document-term matrix.

```r
dtm <- DocumentTermMatrix(corpus) %>% as.matrix()
```

Step 2: Preprocessing, Tokenizing and Converting to DTM (tidy)

With `tidy` we can accomplish the same task as follows.

```r
dtm <- df %>%
  mutate(doc_id = row_number()) %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words) %>%
  mutate(word = stemDocument(word)) %>%
  group_by(doc_id) %>%
  count(word) %>%
  cast_dtm(doc_id, word, n) %>%
  as.matrix()
```

```
## Joining, by = "word"
```

Step 3: Creating a distance matrix

Euclidean Distance

```
euc.dis <- dist(dtm)
euc.dis.mat <- as.matrix(euc.dis)
```

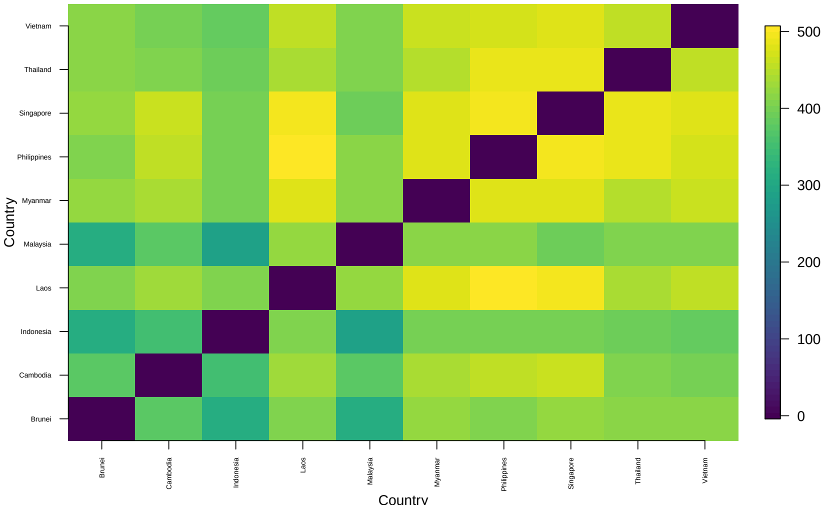
Cosine Distance

```
cos.dis <- dist(dtm, method="cosine")
cos.dis.mat <- as.matrix(cos.dis)
```

Step 4: Visualization

Euclidean Distance

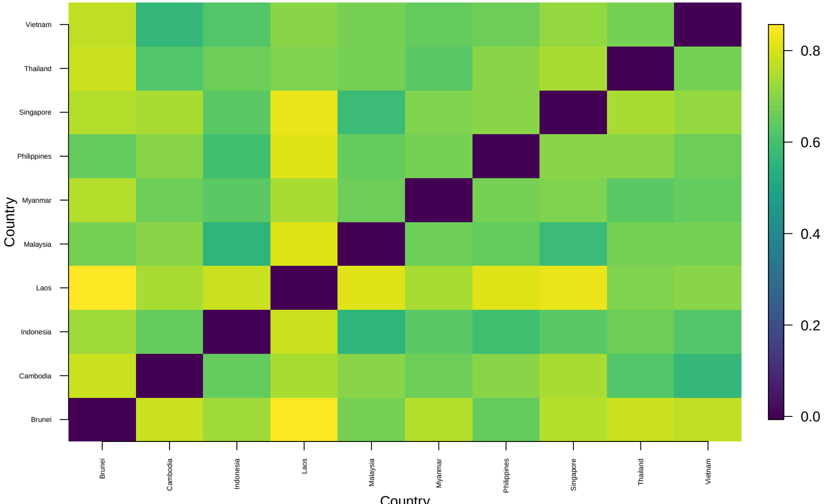
```r
image.plot(1:ncol(euc.dis.mat), 1:ncol(euc.dis.mat), euc.dis.mat, axes = F, xlab="Country", ylab="Country'
axis(1, 1:ncol(euc.dis.mat), df$country_names, cex.axis = 0.5, las=3)
axis(2, 1:ncol(euc.dis.mat), df$country_names, cex.axis = 0.5, las=1)
```



Step 4: Visualization

Cosine Distance

```r
image.plot(1:ncol(cos.dis.mat), 1:ncol(cos.dis.mat), cos.dis.mat, axes = F, xlab="Country", ylab="Country"
axis(1, 1:ncol(cos.dis.mat), df$country_names, cex.axis = 0.5, las=3)
axis(2, 1:ncol(cos.dis.mat), df$country_names, cex.axis = 0.5, las=1)
```



K-Means Clustering

Clustering Wine Types

The dataset `wine.csv` contains the result of a chemical analysis of Italian wines drawn from three different varieties. The dataset has 178 observations.

The variables include: Alcohol, Malic acid, Ash, Alcalinity, Magnesium, Phenols, Flavanoids, Nonflavanoid, Proanthocyanins, Color intensity, Hue, Dilution, and Proline.

The goal of our analysis is to discern underlying patterns, identify similar observations given the variables and determine the number of clusters, using k-means.

Step 1: Importing the data

```
wine <- read.csv("wine.csv", header=T, stringsAsFactors = F
wine$Type <- as.factor(wine$Type)
knitr::kable(head(wine))
```

X	Type	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	F
1	1	14.23	1.71	2.43	15.6	127	2.80	
2	1	13.20	1.78	2.14	11.2	100	2.65	
3	1	13.16	2.36	2.67	18.6	101	2.80	
4	1	14.37	1.95	2.50	16.8	113	3.85	
5	1	13.24	2.59	2.87	21.0	118	2.80	
6	1	14.20	1.76	2.45	15.2	112	3.27	

Step 2: Determine the number of unique wine types

```r
k.value <- n_distinct(wine$Type)
k.value
```

```
## [1] 3
```

Step 3: Use k-means to cluster the wines, using all relevant variables (k = 3)

```r
set.seed(12)
kml <- kmeans(select(wine, -c(X, Type)),
              k.value, nstart = 15)
```

Step 4: Table & Visualization

```r
wine$cluster <- as.factor(kml$cluster)
table(wine$Type, wine$cluster)
```

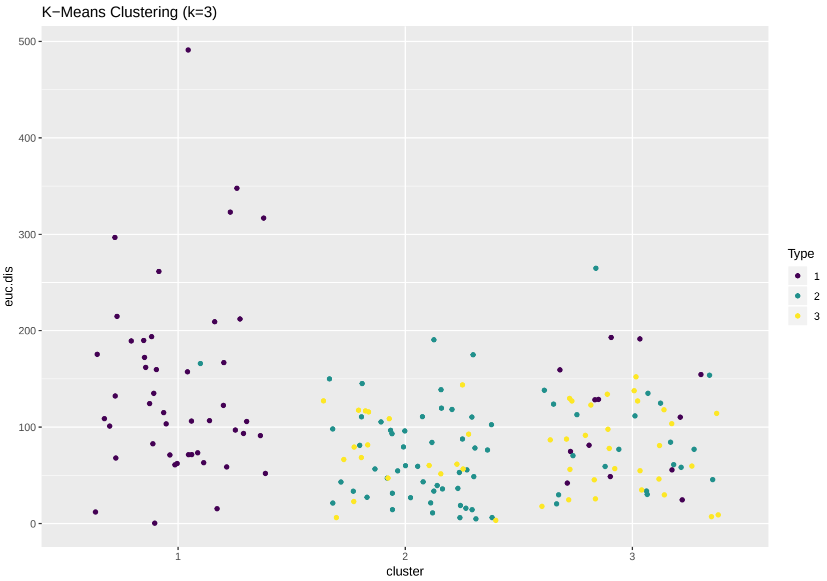
```
##
##       1  2  3
##    1 46  0 13
##    2  1 50 20
##    3  0 19 29
```

Step 4: Table & Visualization

```r
wine$euc.dis <- sqrt(rowSums(select(wine, -c(X, Type, cluster)) -
                               fitted(kml))^ 2)

ggplot(wine, aes(x = cluster, y = euc.dis, col = Type)) +
  geom_jitter() +
  scale_colour_viridis_d()
```

Step 4: Table & Visualization



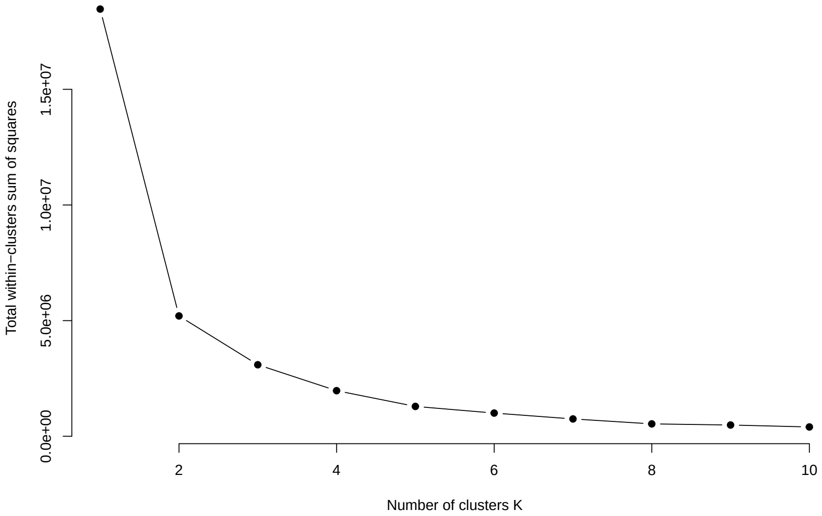
Step 5: Determining the Optimal Number of Clusters

Often, we do not know the number of clusters in advance. In this case, we need to run the model several times, varying the value of k each time in order to find the value of k beyond which the quality of the model is no longer improving as much as the complexity of the model.

```
k.max <- 10
wss <- NULL
for(k in 1:k.max){
  mod <- kmeans(select(wine, -c(X, Type, cluster)), k, nstart = 15)
  wss[k] <- mod$tot.withinss
}
```

Step 5: Determining the Optimal Number of Clusters

```
plot(1:k.max, wss,
     type = "b", pch = 19, frame = FALSE,
     xlab = "Number of clusters K",
     ylab = "Total within-clusters sum of squares")
```



EM

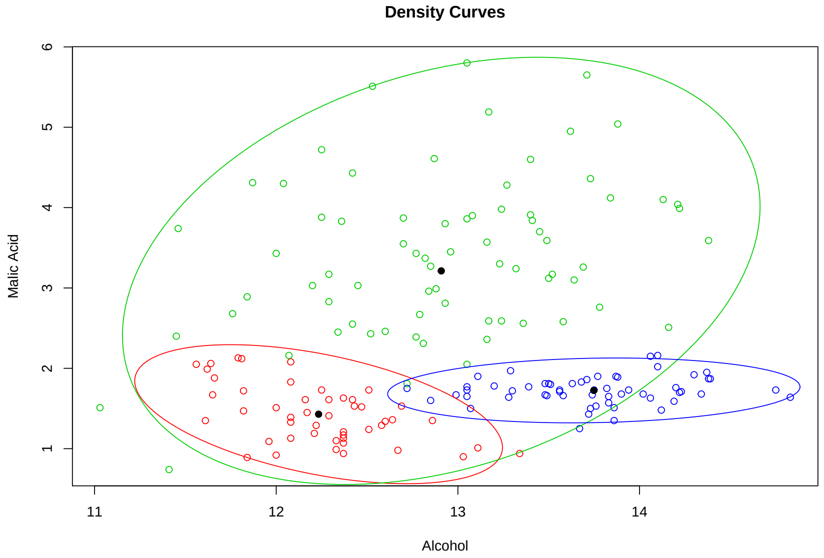
Clustering Wine Types using Two Variables

```r
mod <- mvnormalmixEM(select(wine, Alcohol, Malic), k=3)
```

```
## number of iterations= 100
```

Visualization

```r
plot(mod, whichplots = 2, xlab2="Alcohol", ylab2="Malic Acid")
```


Density Curves
Malic Acid
Alcohol