Assignment 1: Text Pre-Processing & Clustering

Stefan Haböck & Bernhard Mayr

19.11.2021

Dependencies

```
library(tm)
library(cluster)
library(factoextra)
library(proxy)
library(dplyr)
library(purrr)
library(NLP)
library(ggplot2)
```

Loading Data

- `strip.white=TRUE` automatically trims the document contents
- `tibble` is the newer DataFrame alternative
- `select(...)` prepares the data for the `tm` -package

```
loaded_lectures <- read.table("./lectures.txt", sep="\t", header=TRUE, strip.white=TRUE) %>%
  tibble::as_tibble() %>%
  select(doc_id = ID, text = Description, title = Title) %>%
  DataframeSource() %>%
  VCorpus()
```

Data Transformation

- first transform the words to lowercase for more equality and because our clustering approach does not differentiate between uppercase and lowercase words
- then apply stopword removal
- and then stem the words

```
prepared_lectures <- loaded_lectures %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(removeWords, stopwords::stopwords("english")) %>%
  tm_map(stemDocument)
```

Document Term Matrix Creation

- first the TFiDF-Matrix is generated using R’s `tm` -package
- then the sparse terms are removed (this reduces the number of terms by 1/5, but keeps nearly all information)
- now the distance matrix is calculated based on the euclidean distance measurement

```
dtm <- DocumentTermMatrix(prepared_lectures)
dtm.tfidf <- dtm %>%
  weightTfIdf() %>%
  removeSparseTerms(0.99)

tfidf.matrix <- as.matrix(dtm.tfidf)
dist.matrix = dist(tfidf.matrix, method = "cosine")
```

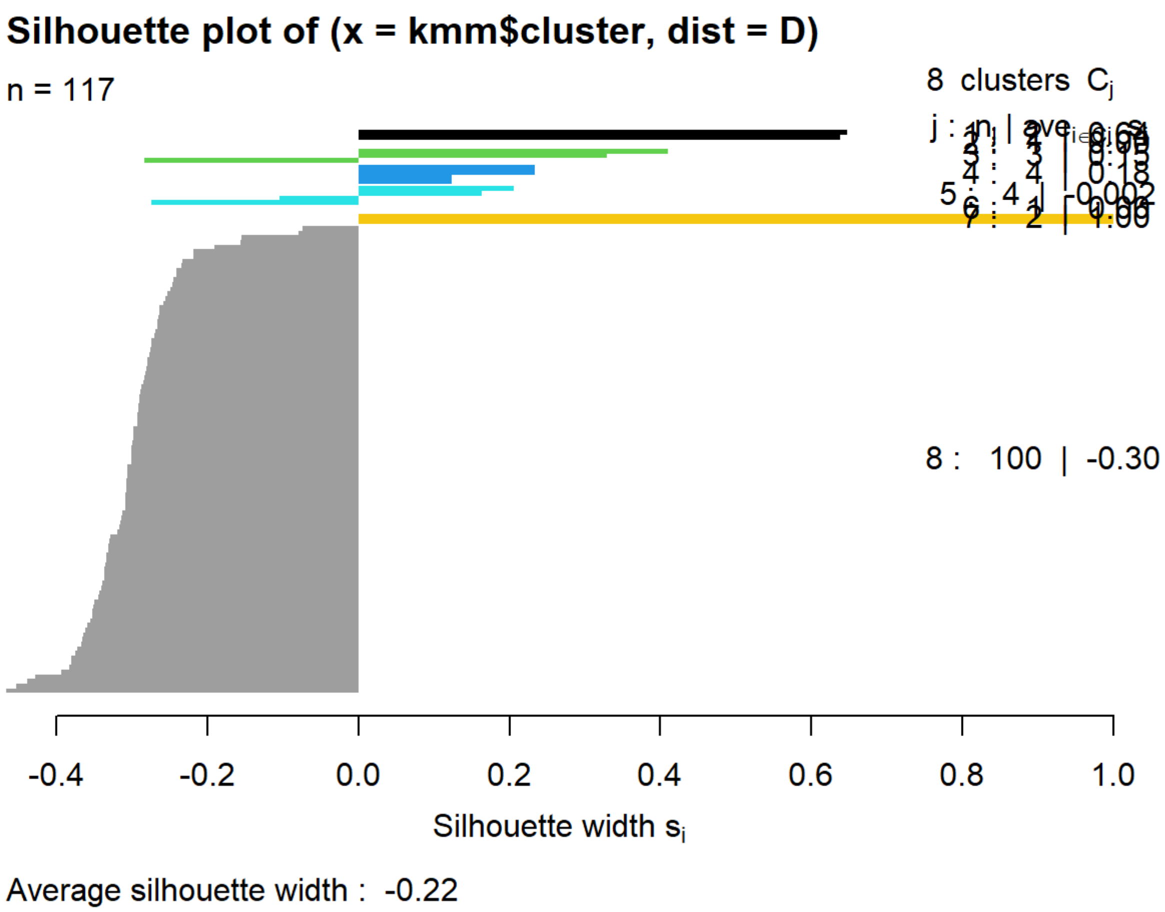
Clustering

- kmeans clustering is calculated based on a K of 8 clusters
- hierarchical clustering is calculated based on the `ward.d2` method

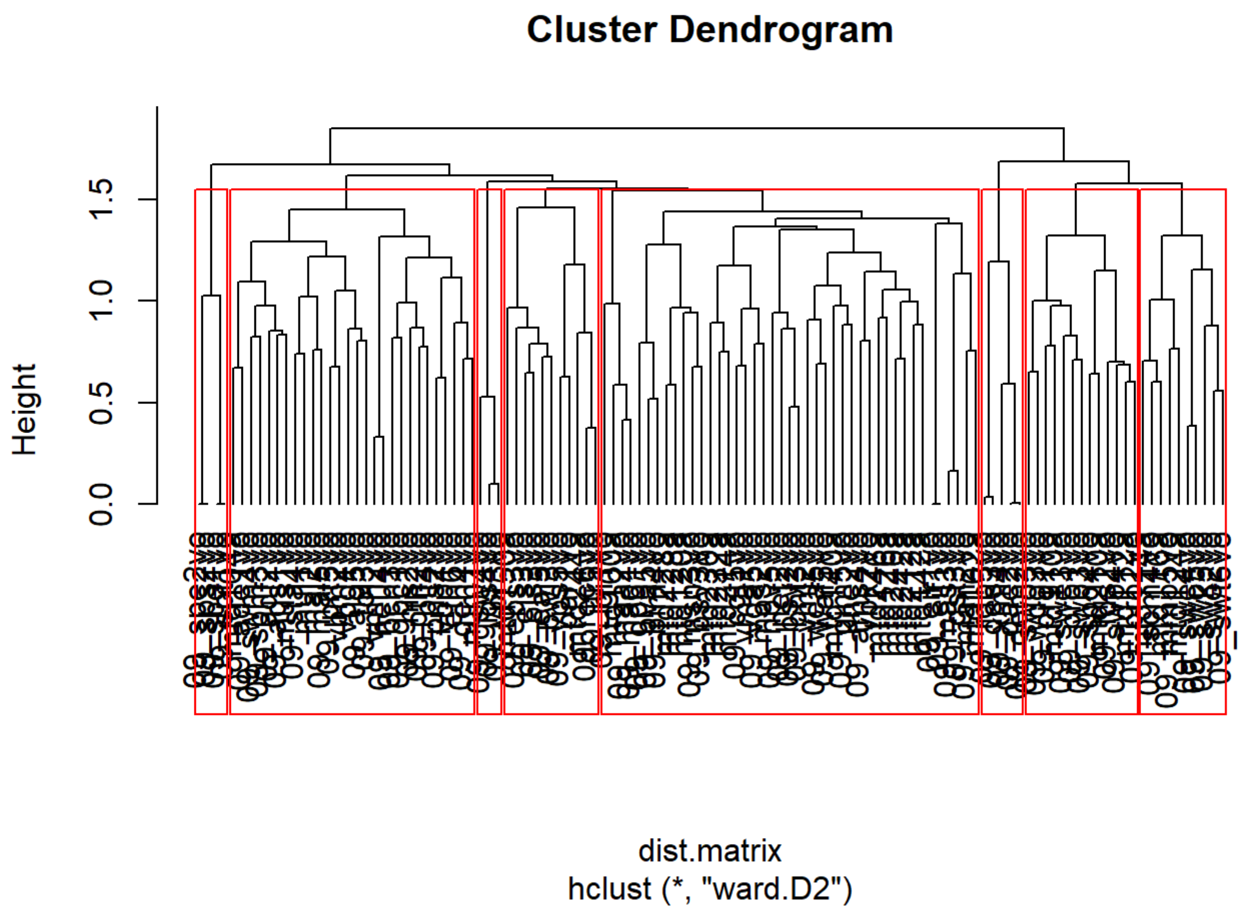
```
truth.K = 8
clustering.kmeans <- kmeans(tfidf.matrix, truth.K)
clustering.hierarchical <- hclust(dist.matrix, method = "ward.D2")

master.cluster <- clustering.kmeans$cluster
slave.hierarchical <- cutree(clustering.hierarchical, k = truth.K)
```

Silhouette and Dendrogram plot

```
# Silhouette Plot
kmm <- kmeans(tfidf.matrix, truth.K)
D <- daisy(tfidf.matrix, metric = "gower")
plot(silhouette(kmm$cluster, D), col=1:truth.K, border=NA)
```



```
# Cluster Dendrogram
plot(clustering.hierarchical, hang = -1)
rect.hclust(clustering.hierarchical, k = truth.K, border = "red")
```



Title and Index mapping

- the titles are mapped to the chart's index data to provide a better understanding of how the data has been clustered

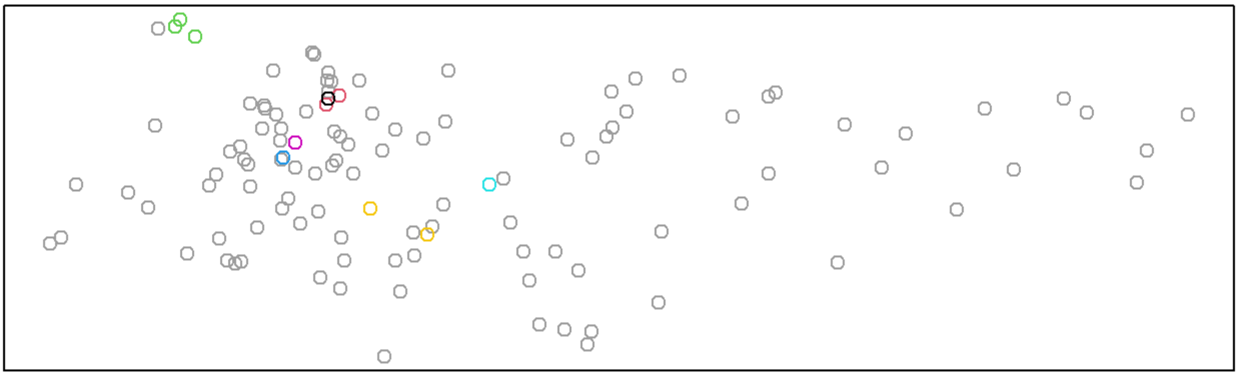
```
#get titles
titles = map(as.list(prepared_lectures), "meta.id")
titles <- names(titles)
# map the titles to the cluster number
kmm_matrix <- cbind(titles, kmm$cluster)
# sort the data based on the cluster numbers
kmm_matrix <- kmm_matrix[order(kmm_matrix[,2]),]

# map the histogram data to the index numbers of the diagram
hist_matrix <- cbind(titles[clustering.hierarchical$order], clustering.hierarchical$order)
```

Cluster Comparison plot

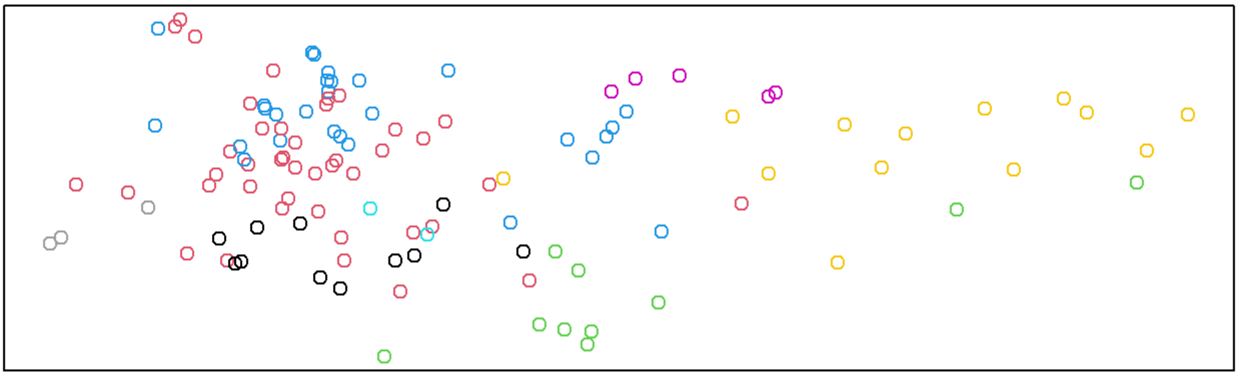
- taken from Text Clustering with R: an Introduction for Data Scientists (https://medium.com/@SAPCAI/text-clustering-with-r-an-introduction-for-data-scientists-c406e7454e76)

```
points <- cmdscale(dist.matrix, k = 2)
palette <- colorspace::diverge_hcl(truth.K) # Creating a color palette
previous.par <- par(mfrow=c(2,1), mar = rep(1.5, 4))

plot(points, main = 'K-Means clustering', col = as.factor(master.cluster),
     mai = c(0, 0, 0, 0), mar = c(0, 0, 0, 0),
     xaxt = 'n', yaxt = 'n', xlab = '', ylab = '')
plot(points, main = 'Hierarchical clustering', col = as.factor(slave.hierarchical),
     mai = c(0, 0, 0, 0), mar = c(0, 0, 0, 0),
     xaxt = 'n', yaxt = 'n', xlab = '', ylab = '')
```

K-Means clustering



Hierarchical clustering



```
par(previous.par) # recovering the original plot space parameters
```

Exercise Answers

- **3c:** Use different values for the number of clusters k. Have a look at the documents combined to a cluster. Which k works best?
    - after trying some values for k, we decided that a cluster amount of 8 worked pretty well
- **3d:** Use different distance measurements combined with different values for k. Which distance works best (with which cluster size)?
    - combinig different cluster sizes and distance measurements, we came to the conclusion that *cosine* worked way better than *manhattan* and *euclidean*, given our k of 8
- **3e:** Perform various runs. Does the result look similar? Does changing the number of iterations have any effect?
    - the results look different, especially for the silhouette plot, this stems from the fact, that the first members of the clusters are choosen randomly

- **4e:** Test various distance measurements and different linkage structures. Which work best? Can you see any peculiarities of the linkage methods?
    - we tried some of the linkage methods and evaluated them based on the distribution of the cluster size, here *ward.d2* clearly worked the best
- **5a:** Use tfidf instead of term occurrence for the TtD. Does it improve the results?
    - TFiDF combined with dropping sparse terms really improved classification in terms of creating cohesive and equally distributed clusters

Resources

- https://mran.microsoft.com/snapshot/2018-03-30/web/packages/tm/vignettes/tm.pdf (https://mran.microsoft.com/snapshot/2018-03-30/web/packages/tm/vignettes/tm.pdf)
- https://cran.r-project.org/web/packages/tm/tm.pdf (https://cran.r-project.org/web/packages/tm/tm.pdf)
- https://medium.com/@SAPCAI/text-clustering-with-r-an-introduction-for-data-scientists-c406e7454e76 (https://medium.com/@SAPCAI/text-clustering-with-r-an-introduction-for-data-scientists-c406e7454e76)
- https://books.psychstat.org/textmining/cluster-analysis.html (https://books.psychstat.org/textmining/cluster-analysis.html)