Simple word clustering

TEXT MINING WITH BAG-OF-WORDS IN R

Ted Kwartler
Instructor

datacamp

Hierarchical clustering example

```
dist_rain <- dist(rain[, 2])
```

The data

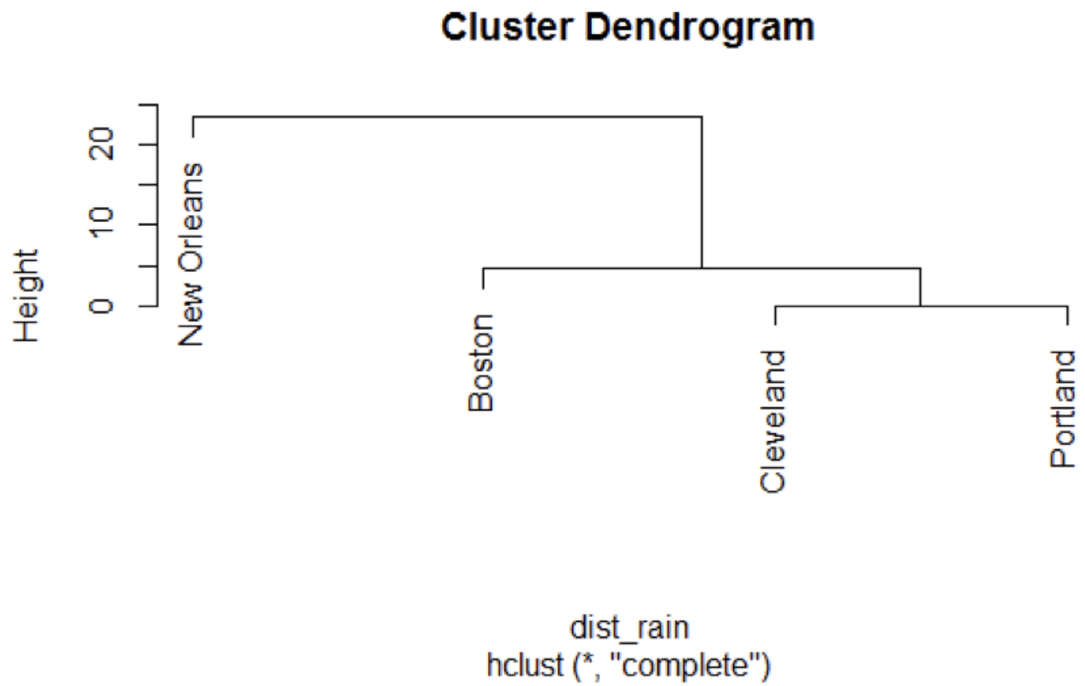
City	Annual rainfall
Cleveland	39.14
Portland	39.14
Boston	43.77
New Orleans	62.45

Distance matrix

	Cleveland	Portland	Boston
Portland	0.00		
Boston	4.63	4.63	
New Orleans	23.31	23.31	18.69

A simple dendrogram

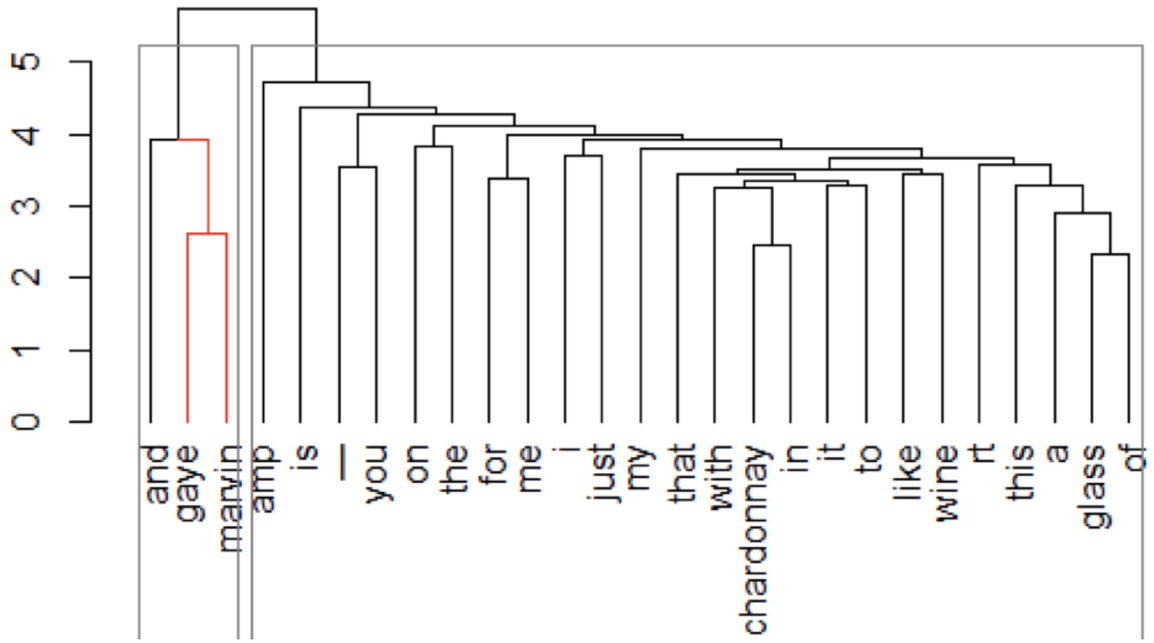
```
# Convert to hierarchical cluster obj
hc <- hclust(dist_rain)
# Plot dendrogram with city labels
plot(hc, labels = rain$city)
```



Dendrogram aesthetics

```
# Load dendextend package
library(dendextend)
# Convert distance matrix to dendrogram
hc <- hclust(tweets_dist)
hcd <- as.dendrogram(hc)
```

```
# Color branches
hcd <- branches_attr_by_labels(hcd,
           c("marvin", "gaye"), "red")
# Plot dendrogram with some aesthetics
plot(hcd, main = "Better Dendrogram")
rect.dendrogram(hcd, k = 2, border = "grey50")
```



Let's practice!

TEXT MINING WITH BAG-OF-WORDS IN R

datacamp

Getting past single words

TEXT MINING WITH BAG-OF-WORDS IN R

Ted Kwartler
Instructor

datacamp

Unigrams, bigrams, trigrams, oh my!

```
# Use only first 2 coffee tweets
tweets$text[1:2]
```

```
[1] @ayyytylerb that is so true drink lots of coffee
[2] RT @bryzy_brib: Senior March tmw morning at 7:25 A.M. in the SENIOR lot. Get up early, make yo coffee/breakfas
```

```
# Make a unigram DTM on first 2 coffee tweets
unigram_dtm <- DocumentTermMatrix(text_corp)
unigram_dtm
```

```
<<DocumentTermMatrix (documents: 2, terms: 18)>>
Non-/sparse entries: 18/18
Sparsity           : 50%
Maximal term length: 15
Weighting          : term frequency (tf)
```

Unigrams, bigrams, trigrams, oh my!

```
# Load RWeka package
library(RWeka)
```

```
# Define bigram tokenizer
tokenizer <- function(x) NGramTokenizer(x, Weka_control(min = 2, max = 2))
# Make a bigram TDM
bigram_tdm <- TermDocumentMatrix(clean_corpus(text_corp),
                                 control = list(tokenize = tokenizer))
bigram_tdm
```

```
<<DocumentTermMatrix (documents: 2, terms: 21)>>
Non-/sparse entries: 21/21
Sparsity           : 50%
Maximal term length: 19
Weighting          : term frequency (tf)
```

Let's practice!

TEXT MINING WITH BAG-OF-WORDS IN R

Different frequency criteria

TEXT MINING WITH BAG-OF-WORDS IN R

Ted Kwartler
Instructor

datacamp

Term weights

- Default term frequency = simple word count
- Frequent words can mask insights
- Adjust term weighting via Tfldf
- Words appearing in many documents are penalized

Term weights

```
# Standard term weighting
tf_tdm <- TermDocumentMatrix(text_corp)
tf_tdm_m <- as.matrix(tf_dtm)
tf_tdm_m[505:510, 5:10]
```

	Docs					
Terms	5	6	7	8	9	10
cocoa	0	0	0	0	0	0
cocobear	0	0	0	0	0	0
coconut	0	0	0	0	0	0
codagogy	0	0	0	0	0	0
code-alan	0	0	0	0	0	0
coffee	1	1	1	1	1	1

```
# TfIdf weighting
tf_idf_tdm <- TermDocumentMatrix(text_corp,
    control = list(weighting = weightTfIdf))
tf_idf_tdm_m <- as.matrix(tf_idf_dtm)
tf_tdm_m <- as.matrix(tf_dtm)
```

	Docs					
Terms	5	6	7	8	9	10
cocoa	0.00	0.000	0.000	0.000	0.000	0.000
cocobear	0.00	0.000	0.000	0.000	0.000	0.000
coconut	0.00	0.000	0.000	0.000	0.000	0.000
codagogy	0.00	0.000	0.000	0.000	0.000	0.000
code-alan	0.00	0.000	0.000	0.000	0.000	0.000
coffee	0.01	0.014	0.008	0.043	0.022	0.029

Retaining document metadata

```
# Ensure the first 2 columns are doc_id & text
names(tweets)[1:2] <- c('doc_id','text')

# Create VCorpus including metadata
test_corpus <- VCorpus(DataframeSource(tweets))
```

```
# Clean and view results
text_corpus <- clean_corpus(text_corpus)
content(text_corpus[[1]])
```

```
$content
[1] "ayyytylerb true drink lots coffee"
```

```
meta(text_corpus[[1]])
```

```
$meta
  id       : 1
  author   : thejennagibson
  date     : 8/9/2013 2:43
  language: en
```

Let's practice!

TEXT MINING WITH BAG-OF-WORDS IN R

datacamp