quanteda Cheat Sheet

Quantitative Analysis of Textual Data

General syntax

- **corpus_*** manage text collections/metadata
- **tokens_*** create/modify tokenized texts
- **dfm_*** create/modify doc-feature matrices
- **fcm_*** work with co-occurrence matrices
- **textstat_*** calculate text-based statistics
- **textmodel_*** fit (un-)supervised models
- **textplot_*** create text-based visualizations

Consistent grammar:

- ***object*()** constructor for the object type
- ***object_verb*()** inputs & returns object type

Extensions

quanteda works well with these companion packages:

- **readtext**: an easy way to read text data
- **spacyr**: NLP using the spaCy library
- **quanteda.corpora**: additional text corpora
- **stopwords**: multilingual stopword lists in R
- **quanteda.[textstats/textmodels/textplots]** text analysis packages

Create a corpus from texts (corpus_*)

Read texts (txt, pdf, csv, doc, docx, json, xml)

```
my_texts <- readtext::readtext("~/link/to/path/*")
```

Construct a corpus from a character vector

```
x <- corpus(data_char_ukimmig2010, text_field = "text")
```

Explore a corpus

```
summary(data_corpus_inaugural, n = 2)
## Corpus consisting of 58 documents, showing 2 documents:
##
##             Text Types Tokens Sentences Year  President FirstName Party
##  1789-Washington   625   1537        23 1789 Washington    George  none
##  1793-Washington    96    147         4 1793 Washington    George  none
```

Extract or add document-level variables

```
party <- data_corpus_inaugural$Party
x$serial_number <- seq_len(ndoc(x))
docvars(x, "serial_number") <- seq_len(ndoc(x)) # alternative
```

Bind or subset corpora

```
corpus(x[1:5]) + corpus(x[7:9])
corpus_subset(x, Year > 1990)
```

Change units of a corpus

```
corpus_reshape(x, to = "sentences")
```

Segment texts on a pattern match

```
corpus_segment(x, pattern, valuetype, extract_pattern = TRUE)
```

Take a random sample of corpus texts

```
corpus_sample(x, size = 10, replace = FALSE)
```

Tokenize a set of texts (tokens_*)

Tokenize texts from a character vector or corpus

```
toks <- tokens("Powerful tool for text analysis.")
```

Convert sequences into compound tokens

```
myseqs <- phrase(c("text analysis"))
tokens_compound(toks, myseqs)
```

Select tokens

```
tokens_select(toks, c("powerful", "text"), selection = "keep")
```

Create a dictionary

```
dict <- dictionary(list(negative = c("bad", "awful", "sad"),
                        positive = c("good", "wonderful", "happy")))
```

Apply a dictionary

```
tokens_lookup(toks, dictionary = data_dictionary_LSD2015)
```

Create ngrams and skipgrams from tokens

```
tokens_ngrams(toks, n = 1:3)
tokens_skipgrams(toks, n = 2, skip = 0:1)
```

Convert case of tokens

```
tokens_tolower(toks) tokens_toupper(toks)
```

Stem tokens

```
tokens_wordstem(toks)
```

tokens_remove/select/toupper/tolower() are also available

Extract features (dfm_*)

Create a document-feature matrix (dfm) from a tokens object

```
dfmat <- dfm(toks)
```

Select features

```
dfm_select(dfmat, pattern = "recommend*"), selection = "keep")
```

Randomly sample documents or features

```
dfm_sample(dfmat, what = c("documents", "features"))
```

Weight or smooth the feature frequencies

```
dfm_weight(dfmat, scheme = "prop")
dfm_smooth(dfmat, smoothing = 0.5)
```

Sort or group a dfm

```
dfm_sort(dfmat, margin = c("features", "documents", "both"))
dfm_group(dfmat, groups = President)
```

Combine identical dimension elements of a dfm

```
dfm_compress(dfmat, margin = c("both", "documents", "features"))
```

Create a feature co-occurrence matrix (fcm)

```
x <- fcm(data_corpus_inaugural, context = "window", size = 5)
```

fcm_compress/remove/select/toupper/tolower() are also available

by **Stefan Müller** and **Kenneth Benoit** • *smueller@quanteda.org, kbenoit@quanteda.org*

Learn more at: https://quanteda.io • updated: 12/2023

Useful additional functions

Locate keywords-in-context

```
kwic(tokens(data_corpus_inaugural), pattern = "america*")
## Keyword-in-context with 499 matches.
## [1789-Washington, 1069] hands of the    | American   | people. Besides
## [1789-Washington, 1472] to favor the    | American   | people with opportunities
## [1793-Washington, 63] people of united  |  America   | . Previous to
## [1797-Adams, 16] middle course for      |  America   | remained between unlimited
```

Utility functions

`as.character(`*corpus*`)`	*Show texts of a corpus*
`ndoc(`*corpus / dfm / tokens*`)`	*Count documents/features*
`nfeat(`*corpus / dfm / tokens*`)`	*Count features*
`ntoken(`*corpus / dfm / tokens*`)`	*Count tokens*
`summary(`*corpus / dfm*`)`	*Print summary*
`head(`*corpus / dfm*`)`	*Return first part*
`tail(`*corpus / dfm*`)`	*Return last part*

Calculate text statistics (`textstat_*`)

These functions require the **quanteda.textstats** *package*

Tabulate feature frequencies from a dfm

```
textstat_frequency(x) topfeatures(x)
```

Identify and score collocations from a tokenized text

```
toks <- tokens(c("quanteda is a pkg for quant text analysis",
                 "quant text analysis is a growing field"))
textstat_collocations(toks, size = 3, min_count = 2)
```

Calculate readability of a corpus

```
textstat_readability(x, measure = c("Flesch", "FOG"))
```

Calculate lexical diversity of a dfm

```
textstat_lexdiv(x, measure = "TTR")
```

Measure distance or similarity from a dfm

```
textstat_simil(x, "2017-Trump", method = "cosine",
               margin = c("documents", "features"))
textstat_dist(x, "2017-Trump",
              margin = c("documents", "features"))
```

Calculate keyness statistics

```
textstat_keyness(x, target = "2017-Trump")
```

Fit text models based on a dfm (`textmodel_*`)

These functions require the **quanteda.textmodels** *package*

Correspondence Analysis (CA)

```
textmodel_ca(x, threads = 2, sparse = TRUE, residual_floor = 0.1)
```

Naïve Bayes classifier for texts

```
textmodel_nb(x, y = training_labels, distribution = "multinomial")
```

SVM classifier for texts

```
textmodel_svm(x, y = training_labels)
```

Wordscores text model

```
refscores <- c(seq(-1.5, 1.5, .75), NA))
textmodel_wordscores(data_dfm_lbgexample, refscores)
```

Wordfish Poisson scaling model

```
textmodel_wordfish(dfm(data_corpus_irishbudget2010), dir = c(6,5))
```

Textmodel methods: `predict()`, `coef()`, `summary()`, `print()`

Plot features or models (`textplot_*`)

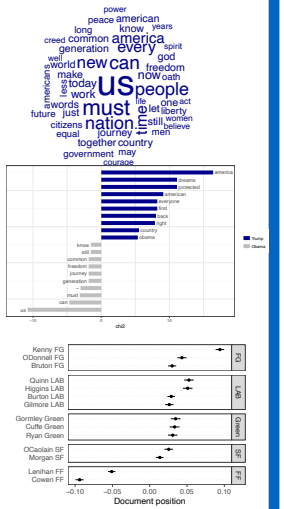
These functions require the **quanteda.textplots** *package*

Plot features as a wordcloud

```
data_corpus_inaugural |>
  corpus_subset(President == "Obama") |>
  tokens() |>
  tokens_remove(pattern = stopwords("en")) |>
  dfm() |>
  textplot_wordcloud()
```

Plot word keyness

```
data_corpus_inaugural |>
  corpus_subset(President %in%
                  c("Obama", "Trump")) |>
  tokens() |>
  dfm() |>
  dfm_group(groups = President) |>
  textstat_keyness(target = "Trump") |>
  textplot_keyness()
```



Plot Wordfish, Wordscores or CA models

```
textplot_scale1d(scaling_model, margin = "documents")
```

Convert dfm to a non-quanteda format

```
convert(x, to = c("lda", "tm", "stm", "austin", "topicmodels",
                  "lsa", "matrix", "data.frame"))
```

by **Stefan Müller** and **Kenneth Benoit**
smueller@quanteda.org, kbenoit@quanteda.org

Learn more at: https://quanteda.io • updated: 12/2023