# Canzone dei Dodici Mesi

## Canzone dei Dodici Mesi di Guccini

As a bonus to the Caparezza's songs analysis, here is an analogue notebook which focuses on the lyrics of "Canzone dei Dodici Mesi" by Francesco Guccini.

```
library(geniusr)
library(tidyverse)
```

```
## Warning: il pacchetto 'lubridate' è stato creato con R versione 4.2.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.0     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.1     ✓ tibble    3.2.0
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflict
s to become errors
```

```
library(tidytext)
```

```
## Warning: il pacchetto 'tidytext' è stato creato con R versione 4.2.3
```

```
library(quanteda)
```

```
## Warning: il pacchetto 'quanteda' è stato creato con R versione 4.2.3
```

```
## Package version: 3.3.1
## Unicode version: 13.0
## ICU version: 69.1
## Parallel computing: 8 of 8 threads used.
## See https://quanteda.io for tutorials and examples.
```

```
library(quanteda.textstats)
```

```
## Warning: il pacchetto 'quanteda.textstats' è stato creato con R versione 4.2.3
```

```
library(udpipe)
```

```
## Warning: il pacchetto 'udpipe' è stato creato con R versione 4.2.3
```

```
library(wordcloud)
```

```
## Warning: il pacchetto 'wordcloud' è stato creato con R versione 4.2.3
```

```
## Caricamento del pacchetto richiesto: RColorBrewer
```

```
library(textdata)
```

```
## Warning: il pacchetto 'textdata' è stato creato con R versione 4.2.3
```

```
library(reshape2)
```

```
## Warning: il pacchetto 'reshape2' è stato creato con R versione 4.2.3
```

```
##
## Caricamento pacchetto: 'reshape2'
##
## Il seguente oggetto è mascherato da 'package:tidyr':
##
##     smiths
```

```
library(igraph)
```

```
## Warning: il pacchetto 'igraph' è stato creato con R versione 4.2.3
```

```
##
## Caricamento pacchetto: 'igraph'
##
## I seguenti oggetti sono mascherati da 'package:lubridate':
##
##     %--%, union
##
## I seguenti oggetti sono mascherati da 'package:dplyr':
##
##     as_data_frame, groups, union
##
## I seguenti oggetti sono mascherati da 'package:purrr':
##
##     compose, simplify
##
## Il seguente oggetto è mascherato da 'package:tidyr':
##
##     crossing
##
## Il seguente oggetto è mascherato da 'package:tibble':
##
##     as_data_frame
##
## I seguenti oggetti sono mascherati da 'package:stats':
##
##     decompose, spectrum
##
## Il seguente oggetto è mascherato da 'package:base':
##
##     union
```

# Get lyrics

```
Sys.setenv(GENIUS_API_TOKEN = "VWN0BhFGVa9kdw9iY1dqwmcgo1LHICGF-U5B89k3h0g7szhSergcUNueRJpfck
aM")

lyrics <- get_lyrics_search(artist_name = "Francesco Guccini",
                            song_title = "Canzone dei dodici mesi")

months <- c("Gennaio", "Febbraio", "Marzo", "Aprile", "Maggio", "Giugno",
            "Luglio", "Agosto", "Settembre", "Ottobre", "Novembre", "Dicembre")

song_months <- c(
  rep(months[1], 4), rep(months[2], 4), rep(months[3], 4), rep(NA, 4),
  rep(months[4], 4), rep(months[5], 4), rep(months[6], 4), rep(NA, 4),
  rep(months[7], 2), rep(months[8], 2), rep(months[9], 4), rep(months[10], 4),
  rep(NA, 4), rep(months[11], 4), rep(months[12], 4), rep(NA, 6)
)

lyrics$month <- song_months

lyrics_by_month <- lyrics %>%
  group_by(month) %>%
  summarise(month_lyrics = paste(line, collapse = " ")) %>%
  arrange(match(month, months))
```

```
lyrics_by_month %>% head()
```

```
## # A tibble: 6 × 2
##   month    month_lyrics
##   <chr>    <chr>
## 1 Gennaio  Viene Gennaio, silenzioso e lieve, un fiume addormentato Fra le cui …
## 2 Febbraio Viene Febbraio e il mondo è a capo chino, ma nei convitti e in piazz…
## 3 Marzo    Cantando, Marzo porta le sue piogge, la nebbia squarcia il velo Port…
## 4 Aprile   Con giorni lunghi, al sonno dedicati, il dolce Aprile viene Quali se…
## 5 Maggio   Ben venga Maggio e il gonfalone amico, ben venga primavera Il nuovo …
## 6 Giugno   Giugno, che sei maturità dell'anno, di te ringrazio Dio In un tuo gi…
```

```
doc_ids <- vector()
for(i in 1:nrow(lyrics_by_month)){
  id <- paste("doc", toString(i), sep = "")
  doc_ids <- doc_ids %>% append(id)
}

lyrics_by_month <- lyrics_by_month %>% mutate(doc_id = doc_ids, .before = 1)
```

# Text pre-processing

```
corpus <- corpus(lyrics_by_month$month_lyrics, docnames = lyrics_by_month$doc_id)
summary(corpus)
```

```
## Corpus consisting of 13 documents, showing 13 documents:
##
##   Text Types Tokens Sentences
##   doc1    37     56         2
##   doc2    40     62         2
##   doc3    39     65         2
##   doc4    41     61         2
##   doc5    36     64         2
##   doc6    42     67         2
##   doc7    22     33         1
##   doc8    22     33         1
##   doc9    34     49         2
##  doc10    37     63         2
##  doc11    42     63         2
##  doc12    42     66         2
##  doc13    34    199         3
```

```
cat(as.character(corpus[1]))
```

```
## Viene Gennaio, silenzioso e lieve, un fiume addormentato Fra le cui rive giace come neve i
l mio corpo malato, il mio corpo malato... Sono distese, lungo la pianura, bianche file di ca
mpi Son come amanti dopo l'avventura, neri alberi stanchi, neri alberi stanchi...
```

```
corpus_tokens <- corpus %>%
  quanteda::tokens(remove_punct = TRUE, remove_numbers = TRUE, remove_symbols = TRUE) %>%
  tokens_tolower()
```

```
txt <- sapply(corpus_tokens, FUN=function(x) paste(x, collapse = "\n"))
udpipe_download_model(language = "italian-isdt", model_dir = "resources/")
```

```
## Downloading udpipe model from https://raw.githubusercontent.com/jwijffels/udpipe.models.u
d.2.5/master/inst/udpipe-ud-2.5-191206/italian-isdt-ud-2.5-191206.udpipe to resources//italia
n-isdt-ud-2.5-191206.udpipe
```

```
##  - This model has been trained on version 2.5 of data from https://universaldependencies.o
rg
```

```
##  - The model is distributed under the CC-BY-SA-NC license: https://creativecommons.org/lic
enses/by-nc-sa/4.0
```

```
##  - Visit https://github.com/jwijffels/udpipe.models.ud.2.5 for model license details.
```

```
##  - For a list of all models and their licenses (most models you can download with this pac
kage have either a CC-BY-SA or a CC-BY-SA-NC license) read the documentation at ?udpipe_downl
oad_model. For building your own models: visit the documentation by typing vignette('udpipe-t
rain', package = 'udpipe')
```

```
## Downloading finished, model stored at 'resources//italian-isdt-ud-2.5-191206.udpipe'
```

```
##       language                                      file_model
## 1 italian-isdt resources//italian-isdt-ud-2.5-191206.udpipe
##
url
## 1 https://raw.githubusercontent.com/jwijffels/udpipe.models.ud.2.5/master/inst/udpipe-ud-2.5-191206/italian-isdt-ud-2.5-191206.udpipe
##   download_failed download_message
## 1           FALSE               OK
```

```
lang_model <- udpipe_load_model(file = "resources/italian-isdt-ud-2.5-191206.udpipe")
outL <- udpipe_annotate(lang_model, x = txt, tokenizer = "vertical", trace = TRUE) %>%
  as.data.frame()
```

```
## 2023-05-31 10:51:59 Annotating text fragment 1/13
## 2023-05-31 10:51:59 Annotating text fragment 2/13
## 2023-05-31 10:51:59 Annotating text fragment 3/13
## 2023-05-31 10:51:59 Annotating text fragment 4/13
## 2023-05-31 10:51:59 Annotating text fragment 5/13
## 2023-05-31 10:51:59 Annotating text fragment 6/13
## 2023-05-31 10:51:59 Annotating text fragment 7/13
## 2023-05-31 10:51:59 Annotating text fragment 8/13
## 2023-05-31 10:51:59 Annotating text fragment 9/13
## 2023-05-31 10:51:59 Annotating text fragment 10/13
## 2023-05-31 10:51:59 Annotating text fragment 11/13
## 2023-05-31 10:51:59 Annotating text fragment 12/13
## 2023-05-31 10:51:59 Annotating text fragment 13/13
```

```
it_stopwords <- readLines("https://raw.githubusercontent.com/stopwords-iso/stopwords-it/master/stopwords-it.txt")
```

```
## Warning in
## readLines("https://raw.githubusercontent.com/stopwords-iso/stopwords-it/master/stopwords-it.txt"):
## riga finale incompleta in
## 'https://raw.githubusercontent.com/stopwords-iso/stopwords-it/master/stopwords-it.txt'
```

```
outL <- outL %>% filter(!(token %in% it_stopwords) & !(lemma %in% it_stopwords))
```

```
outL %>% select(doc_id, token, lemma, upos) %>% sample_n(5)
```

```
##   doc_id   token   lemma upos
## 1   doc1   campi   campo NOUN
## 2  doc13 giocare giocare VERB
## 3   doc1  malato  malato  ADJ
## 4  doc13 giocare giocare VERB
## 5   doc3   porta portare VERB
```

```
outL_reduced <- outL %>% filter(upos %in% c("NOUN", "PROPN", "ADJ", "VERB"))
```

```
lemmatized_lyrics <- outL_reduced %>% group_by(doc_id = fct_inorder(doc_id)) %>%
  summarise(lemmatized = paste(lemma, collapse = " "))
lyrics_by_month <- lyrics_by_month  %>% right_join(lemmatized_lyrics, by = "doc_id")

corpus <- lyrics_by_month$lemmatized %>% corpus(docnames = lyrics_by_month$doc_id)
```

```
DTM <- corpus %>% tokens() %>% dfm()

DTM
```

```
## Document-feature matrix of: 13 documents, 197 features (91.53% sparse) and 0 docvars.
##       features
## docs   venire gennaio silenzioso lieve fiume addormentato riva giacere neve
##   doc1      1       1          1     1     1            1    1       1    1
##   doc2      1       0          0     0     0            0    0       0    0
##   doc3      0       0          0     0     0            0    0       0    1
##   doc4      1       0          0     0     0            0    0       0    0
##   doc5      4       0          0     0     0            0    0       0    0
##   doc6      0       0          0     0     0            0    0       0    0
##       features
## docs   corpo
##   doc1     2
##   doc2     0
##   doc3     0
##   doc4     0
##   doc5     0
##   doc6     0
## [ reached max_ndoc ... 7 more documents, reached max_nfeat ... 187 more features ]
```

# Lexical Analysis

```
DTM %>% dim()
```

```
## [1]  13 197
```

```
words <- colnames(DTM)
freqs <- colSums(DTM)
wordlist <- data.frame(words, freqs)
wordlist %>% arrange(-freqs) %>% head()
```

```
##           words freqs
## sapere   sapere    13
## giocare giocare    12
## venire   venire     7
## mano       mano     5
## sole       sole     4
## nascere nascere     4
```

# Data visualization

```
par(mar=c(1,1,0.5,1))
wordcloud(words = wordlist$words, freq = wordlist$freqs, scale = c(3.5, 0.35), max.words = 5
0, random.order = F,
          colors = RColorBrewer::brewer.pal(name = "Dark2", n = 4))
text(0.5, 1, "wordcloud with TF ponderation", font = 2)
```


# TF-IDF ponderation

```
tf_idf <- dfm_tfidf(DTM)
freqs_tf_idf <- colSums(tf_idf)
words_tf_idf <- colnames(tf_idf)
wordlist_tf_idf <- data.frame(words = words_tf_idf, freqs = freqs_tf_idf)
wordlist_tf_idf %>% arrange(-freqs) %>% head(10)
```

```
##           words     freqs
## giocare giocare 13.367320
## sapere   sapere 10.567874
## andare   andare  4.455773
## simile   simile  4.455773
## tarocco tarocco  4.455773
## mano       mano  4.064567
## venire   venire  3.583184
## nascere nascere  3.251653
## sole       sole  2.547288
## malato   malato  2.438740
```

```
par(mar=c(1,1,0.2,1))
wordcloud(words = wordlist_tf_idf$words, freq = wordlist_tf_idf$freqs,
          scale = c(3.5, 0.35), max.words = 50, random.order = F,
          colors = RColorBrewer::brewer.pal(name = "Dark2", n = 4))
text(0.5, 1, "wordcloud with TF-IDF ponderation", font = 2)
```

**wordcloud with TF-IDF ponderation**


# Co-occurrence analysis

```
binDTM <- DTM %>% dfm_weight("boolean")
coocCounts <- t(binDTM) %*% binDTM
as.matrix(coocCounts[16:18, 16:18])
```

```
##             son amante l'avventura
## son           1      1           1
## amante        1      1           1
## l'avventura   1      1           1
```

```
coocTerm <- "sole"
k <- nrow(binDTM)
ki <- sum(binDTM[, coocTerm])
kj <- colSums(binDTM)
names(kj) <- colnames(binDTM)
kij <- coocCounts[coocTerm, ]

mutualInformationSig <- log(k * kij / (ki * kj))
mutualInformationSig <- mutualInformationSig[order(mutualInformationSig, decreasing = TRUE)]

dicesig <- 2 * kij / (ki + kj)
dicesig <- dicesig[order(dicesig, decreasing=TRUE)]

logsig <- 2 * ((k * log(k)) - (ki * log(ki)) - (kj * log(kj)) + (kij * log(kij))
               + (k - ki - kj + kij) * log(k - ki - kj + kij)
               + (ki - kij) * log(ki - kij) + (kj - kij) * log(kj - kij)
               - (k - ki) * log(k - ki) - (k - kj) * log(k - kj))
logsig <- logsig[order(logsig, decreasing=T)]

resultOverView <- data.frame(
  names(sort(kij, decreasing=T)[1:10]), sort(kij, decreasing=T)[1:10],
  names(mutualInformationSig[1:10]), mutualInformationSig[1:10],
  names(dicesig[1:10]), dicesig[1:10],
  names(logsig[1:10]), logsig[1:10],
  row.names = NULL)
colnames(resultOverView) <- c("Freq-terms", "Freq", "MI-terms", "MI", "Dice-Terms", "Dice",
"LL-Terms", "LL")
print(resultOverView)
```

```
##    Freq-terms Freq  MI-terms       MI Dice-Terms      Dice  LL-Terms        LL
## 1        sole    3  febbraio 1.466337       sole 1.0000000    venire 2.2211525
## 2      venire    2     capo  1.466337     venire 0.5714286    malato 0.8416541
## 3      malato    1    chare  1.466337   febbraio 0.5000000  lasciare 0.8416541
## 4    febbraio    1  convitto 1.466337       capo 0.5000000 l'inverno 0.8416541
## 5        capo    1   piazza  1.466337      chare 0.5000000  apparire 0.8416541
## 6       chare    1   dolore  1.466337   convitto 0.5000000     poeta 0.8416541
## 7    convitto    1   vedere  1.466337     piazza 0.5000000     bello 0.8416541
## 8      piazza    1 arleccare 1.466337     dolore 0.5000000     terra 0.8416541
## 9    lasciare    1 carnevale 1.466337     vedere 0.5000000   nascere 0.8416541
## 10     dolore    1 impazzare 1.466337  arleccare 0.5000000      mano 0.8416541
```

# Co-occurrence visualization

```
source("resources/calculateCoocStatistics.R")
numberOfCoocs <- 10
coocTerm <- "sapere"
coocs <- calculateCoocStatistics(coocTerm, binDTM, measure="LOGLIK")
```

```
## Caricamento del pacchetto richiesto: Matrix
```

```
##
## Caricamento pacchetto: 'Matrix'
```

```
## I seguenti oggetti sono mascherati da 'package:tidyr':
##
##     expand, pack, unpack
```

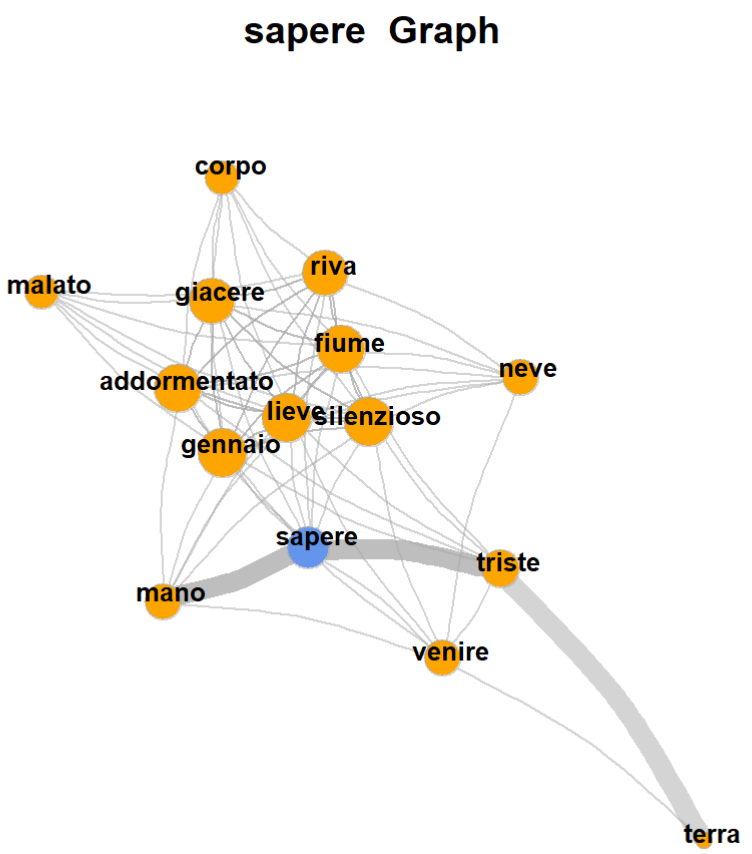
```
print(coocs[1:numberOfCoocs])
```

```
##         mano       triste       venire      gennaio   silenzioso        lieve
##     1.687816     1.687816          NaN          NaN          NaN          NaN
##        fiume addormentato         riva      giacere
##          NaN          NaN          NaN          NaN
```

```
resultGraph <- data.frame(from = character(), to = character(), sig = numeric(0))
tmpGraph <- data.frame(from = character(), to = character(), sig = numeric(0))

# Fill the data.frame to produce the correct number of lines
tmpGraph[1:numberOfCoocs, 3] <- coocs[1:numberOfCoocs]
# Entry of the search word into the first column in all lines
tmpGraph[, 1] <- coocTerm
# Entry of the co-occurrences into the second column of the respective line
tmpGraph[, 2] <- names(coocs)[1:numberOfCoocs]
# Set the significances
tmpGraph[, 3] <- coocs[1:numberOfCoocs]

# Attach the triples to resultGraph
resultGraph <- rbind(resultGraph, tmpGraph)

# Iteration over the most significant numberOfCoocs co-occurrences of the search term
for (i in 1:numberOfCoocs){

  # Calling up the co-occurrence calculation for term i from the search words co-occurrences
  newCoocTerm <- names(coocs)[i]
  coocs2 <- calculateCoocStatistics(newCoocTerm, binDTM, measure="LOGLIK")

  #print the co-occurrences
  coocs2[1:10]

  # Structure of the temporary graph object
  tmpGraph <- data.frame(from = character(), to = character(), sig = numeric(0))
  tmpGraph[1:numberOfCoocs, 3] <- coocs2[1:numberOfCoocs]
  tmpGraph[, 1] <- newCoocTerm
  tmpGraph[, 2] <- names(coocs2)[1:numberOfCoocs]
  tmpGraph[, 3] <- coocs2[1:numberOfCoocs]

  #Append the result to the result graph
  resultGraph <- rbind(resultGraph, tmpGraph[2:length(tmpGraph[, 1]), ])
}

# Sample of some examples from resultGraph
resultGraph[sample(nrow(resultGraph), 6), ]
```

```
##      from           to       sig
## 91   mano        fiume       NaN
## 79   riva      giacere       NaN
## 43 venire    l'inverno 0.3847397
## 87  fiume         neve       NaN
## 10 sapere      giacere       NaN
## 57  fiume addormentato       NaN
```

```r
# set seed for graph plot
set.seed(1)

# Create the graph object as undirected graph
graphNetwork <- graph.data.frame(resultGraph, directed = F)

# Identification of all nodes with less than 2 edges
verticesToRemove <- V(graphNetwork)[degree(graphNetwork) < 2]
# These edges are removed from the graph
graphNetwork <- delete.vertices(graphNetwork, verticesToRemove)

# Assign colors to nodes (search term blue, others orange)
V(graphNetwork)$color <- ifelse(V(graphNetwork)$name == coocTerm, 'cornflowerblue', 'orange')

# Set edge colors
E(graphNetwork)$color <- adjustcolor("DarkGray", alpha.f = .5)
# scale significance between 1 and 10 for edge width
E(graphNetwork)$width <- scales::rescale(E(graphNetwork)$sig, to = c(1, 10))

# Set edges with radius
E(graphNetwork)$curved <- 0.15
# Size the nodes by their degree of networking (scaled between 5 and 15)
V(graphNetwork)$size <- scales::rescale(log(degree(graphNetwork)), to = c(5, 15))

# Define the frame and spacing for the plot
par(mai=c(0,0,1,0))

# Final Plot
plot(
  graphNetwork,
  layout = layout.fruchterman.reingold, # Force Directed Layout
  main = paste(coocTerm, ' Graph'),
  vertex.label.family = "sans",
  vertex.label.cex = 0.8,
  vertex.shape = "circle",
  vertex.label.dist = 0.5,          # Labels of the nodes moved slightly
  vertex.frame.color = adjustcolor("darkgray", alpha.f = .5),
  vertex.label.color = 'black',     # Color of node names
  vertex.label.font = 2,            # Font of node names
  vertex.label = V(graphNetwork)$name,      # node names
  vertex.label.cex = 1 # font size of node names
)
```



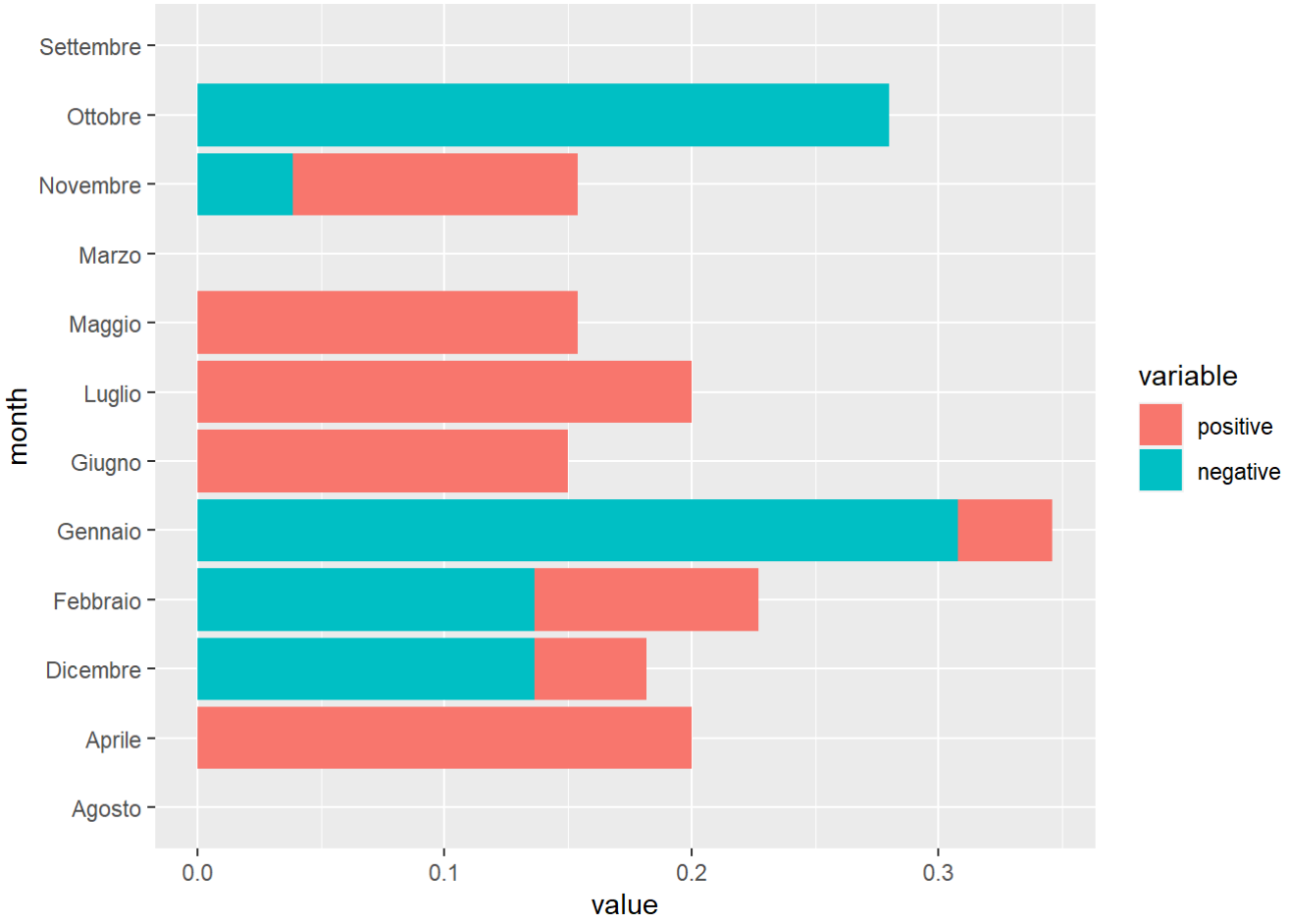

# Sentiment analysis

```
sentiment_lexicon <- read.table("resources/Italian-NRC-EmoLex.txt",
                                header = TRUE, sep = "\t")
sentiment_lexicon_corpus <- sentiment_lexicon %>% filter(Italian.Word %in% colnames(DTM))
positive_terms <- sentiment_lexicon_corpus %>%
  filter(positive == 1) %>%
  select(Italian.Word) %>% pull()
negative_terms <- sentiment_lexicon_corpus %>%
  filter(negative == 1) %>%
  select(Italian.Word) %>% pull()
counts_positive <- rowSums(DTM[, positive_terms])
counts_negative <- rowSums(DTM[, negative_terms])
counts_all_terms <- rowSums(DTM)
relative_sentiment_frequencies <- data.frame(
  positive = counts_positive / counts_all_terms,
  negative = counts_negative / counts_all_terms
)
```

```
sentiments_by_month <- aggregate(relative_sentiment_frequencies,
                                 by = list(month = lyrics_by_month$month), mean)

sentiments_by_month %>% head()
```

```
##      month   positive  negative
## 1   Agosto 0.00000000 0.0000000
## 2   Aprile 0.20000000 0.0000000
## 3 Dicembre 0.04545455 0.1363636
## 4 Febbraio 0.09090909 0.1363636
## 5  Gennaio 0.03846154 0.3076923
## 6   Giugno 0.15000000 0.0000000
```

```
df_sentiment <- melt(sentiments_by_month, id.vars = "month")
ggplot(data = df_sentiment, aes(x = month, y = value, fill = variable)) +
  geom_bar(stat="identity", position="stack") + coord_flip()
```




```
positive_months <- aggregate(
  relative_sentiment_frequencies, by = list(month = lyrics_by_month$month),
  mean) %>% filter(positive > negative)
positive_months
```

```
##      month  positive   negative
## 1   Aprile 0.2000000 0.00000000
## 2   Giugno 0.1500000 0.00000000
## 3   Luglio 0.2000000 0.00000000
## 4   Maggio 0.1538462 0.00000000
## 5 Novembre 0.1153846 0.03846154
```

```
negative_months <- aggregate(
  relative_sentiment_frequencies, by = list(month = lyrics_by_month$month),
  mean) %>% filter(negative > positive)
negative_months
```

```
##      month   positive  negative
## 1 Dicembre 0.04545455 0.1363636
## 2 Febbraio 0.09090909 0.1363636
## 3  Gennaio 0.03846154 0.3076923
## 4  Ottobre 0.00000000 0.2800000
```

```
neutral_months <- aggregate(
  relative_sentiment_frequencies, by = list(month = lyrics_by_month$month),
  mean) %>% filter(positive == negative)
neutral_months
```

```
##       month positive negative
## 1    Agosto        0        0
## 2     Marzo        0        0
## 3 Settembre        0        0
```

# Emotion analysis

```
anger_terms <- sentiment_lexicon_corpus %>%
  filter(anger == 1) %>%
  select(Italian.Word) %>% pull()
fear_terms <- sentiment_lexicon_corpus %>%
  filter(fear == 1) %>%
  select(Italian.Word) %>% pull()
joy_terms <- sentiment_lexicon_corpus %>%
  filter(joy == 1) %>%
  select(Italian.Word) %>% pull()
sadness_terms <- sentiment_lexicon_corpus %>%
  filter(sadness == 1) %>%
  select(Italian.Word) %>% pull()

counts_anger <- rowSums(DTM[, anger_terms])
counts_fear <- rowSums(DTM[, fear_terms])
counts_joy <- rowSums(DTM[, joy_terms])
counts_sadness <- rowSums(DTM[, sadness_terms])

relative_emotion_frequencies <- data.frame(
  anger = counts_anger / counts_all_terms,
  fear = counts_fear / counts_all_terms,
  joy = counts_joy / counts_all_terms,
  sadness = counts_sadness / counts_all_terms
)
```

```
emotions_by_month <- aggregate(relative_emotion_frequencies,
                               by = list(month = lyrics_by_month$month), mean)

head(emotions_by_month)
```

```
##      month      anger       fear        joy    sadness
## 1   Agosto 0.00000000 0.00000000 0.00000000 0.00000000
## 2   Aprile 0.00000000 0.00000000 0.15000000 0.00000000
## 3 Dicembre 0.09090909 0.09090909 0.00000000 0.09090909
## 4 Febbraio 0.00000000 0.09090909 0.04545455 0.13636364
## 5  Gennaio 0.00000000 0.15384615 0.03846154 0.23076923
## 6   Giugno 0.00000000 0.00000000 0.15000000 0.05000000
```

```
df_emotions <- melt(emotions_by_month, id.vars = "month")
ggplot(data = df_emotions, aes(x = month, y = value, fill = variable)) +
  geom_bar(stat="identity", position="stack") + coord_flip()
```