Data cleaning and Preliminar analysis

Riccardo Ruta

5/2022

Contents

Part I: Data cleaning **1**

1) First import the dataset and check variables . . . 1
2) Adjust date.time format . . . 2
3) Create the week variable . . . 2
4) Remove the rows with missing tweets . . . 4
5) Check that the variables make sense . . . 4
6) Create a new dataset with only necessary informations . . . 6
7) Create the corpus . . . 6
8) Create the DFM . . . 6
9) Trim the data . . . 6
10) Remove the emoji . . . 7
11) Weight the feature frequencies in the dfm . . . 7

Part II: Preliminar analysis **8**

1) Topfeatures frquency . . . 8
2) Most common hashtag . . . 11
3) Most frequently mentioned usernames . . . 14

Part I: Data cleaning

1) First import the dataset and check variables

```
# import the data
tw <-  read_csv("data/large_files/politicians_final_corrected.csv", show_col_types = FALSE )

kable(colnames(tw), col.names = "variables")
```

variables
tw_screen_name
nome
tweet_testo
creato_il
creato_il_code
url
party_id
genere
chamber
status

2) Adjust date.time format

```
# RUN IN THIS ORDER !!
Sys.setlocale("LC_TIME", "C")
tw$date <- as.Date(strptime(tw$creato_il,"%a %b %d %H:%M:%S %z %Y", tz = "CET"))
tw$date <- na.replace(tw$date, as.Date(tw$creato_il))
```

Check the conversion

```
check_dates <- tw %>% select(creato_il,date)
kable(head(check_dates), col.names = c("Old date", "New date"))
```

Old date	New date
2021-02-13	2021-02-13
2021-02-09	2021-02-09
2021-02-07	2021-02-07
2021-01-21	2021-01-21
2021-01-21	2021-01-21
2021-01-20	2021-01-20

```
kable(tail(check_dates), col.names = c("Old date", "New date"))
```

Old date	New date
Mon Dec 28 09:51:35 +0000 2020	2020-12-28
Tue Jul 20 11:15:44 +0000 2021	2021-07-20
Thu Nov 26 13:46:51 +0000 2020	2020-11-26
Fri Oct 15 17:28:57 +0000 2021	2021-10-15
Wed Jun 03 12:22:31 +0000 2020	2020-06-03
Fri Dec 03 21:01:20 +0000 2021	2021-12-03

3) Create the week variable

```
tw <- tw %>% mutate(week = cut.Date(date, breaks = "1 week", labels = FALSE))
```

```
max(tw$date)
```

Inspect the first and the last dates and check if the number of weeks is correct

```
## [1] "2022-04-18"
```

```
min(tw$date)
```

```
## [1] "2020-01-01"
```

```
difftime(max(tw$date), min(tw$date), units = "weeks")
```

```
## Time difference of 119.7143 weeks
```

Create the month variable

```
tw <- tw %>% mutate(month = cut.Date(date, breaks = "1 month", labels = FALSE))
```

```
max(tw$month)
```

Check the number of month

```
## [1] 28
```

```
length(seq(from = min(tw$date), to = max(tw$date), by = 'month'))
```

```
## [1] 28
```

Count the number of missing values

```
sum(is.na(tw))
```

```
## [1] 154672
```

Inspect where are those missings

```r
missings <- c(
sum(is.na(tw$tw_screen_name)),
sum(is.na(tw$nome)),
sum(is.na(tw$tweet_testo)),
sum(is.na(tw$creato_il)),
sum(is.na(tw$creato_il_code)),
sum(is.na(tw$url)),
sum(is.na(tw$party_id)),
sum(is.na(tw$genere)),
sum(is.na(tw$chamber)),
sum(is.na(tw$status)),
sum(is.na(tw$date)),
sum(is.na(tw$week)),
sum(is.na(tw$month)) )

missing_df <- data.frame(colnames(tw), missings)
kable(missing_df)
```

colnames.tw.	missings
tw_screen_name	0
nome	0
tweet_testo	6494
creato_il	0
creato_il_code	0
url	148178
party_id	0
genere	0
chamber	0
status	0
date	0
week	0
month	0

From that analysis i obtain 148178 url missing, this is because the url is collected only when the tweets has an external link to other sources, for our analysis we can ignore those missings, with this check also results 6494 tweets missing those are the cases when someone post only images or video without text, so the extraction is correct.

4) Remove the rows with missing tweets

```r
sum(is.na(tw$tweet_testo))
```

```
## [1] 6494
```

```r
tw <- tw %>% drop_na(tweet_testo)
```

5) Check that the variables make sense

```r
unique(tw$party_id)
```

```
##  [1] "PD"           "FDI"          "M5S"          "FI"           "REG_LEAGUES"
##  [6] "MISTO"        "LEGA"         "IV"           "INDIPENDENTE" "CI"
## [11] "LEU"
```

```r
unique(tw$genere)
```

```
## [1] "male"   "female" "male "
```

```r
unique(tw$chamber)
```

```
## [1] "NotParl" "Senate"  "Camera"
```

```r
unique(tw$status)
```

```
## [1] "sottosegretario" "presregione"     "viceministro"    "ministro"
## [5] "segretario"      "Parl"
```

The variable genere needs to be corrected

```r
# Remove space from genere variable [RUN ONLY ONCE!]
a <- unique(tw$genere)
a[3]
```

```
## [1] "male "
```

```r
which(tw$genere == a[3])
```

```
## [1] 33300 33301 33302 33303 33304
```

```r
tw$genere <- gsub(a[3],"male",tw$genere)
```

Check the substitution

```r
which(tw$genere == a[3])
```

```
## integer(0)
```

```r
unique(tw$genere)
```

```
## [1] "male"   "female"
```

Now all the variables are ready for next steps

6) Create a new dataset with only necessary informations

```
# Select variables for the analysis
dataset <- tw %>% select(nome, tweet_testo, genere, party_id,chamber,status, date, week, month )
colnames(dataset)
```

```
## [1] "nome"        "tweet_testo" "genere"      "party_id"    "chamber"
## [6] "status"      "date"        "week"        "month"
```

7) Create the corpus

```
corpus <- corpus(dataset, text = "tweet_testo")
ndoc(corpus)
```

```
## [1] 391197
```

8) Create the DFM

```
# Split the corpus into single tokens (remain positional)
doc.tokens <- tokens(corpus,
                                 remove_punct = TRUE,
                                 remove_numbers = TRUE,
                                 remove_symbols = TRUE,
                                 remove_url = TRUE)

# Import my stopwords
my_word <- as.list(read_csv("data/it_stopwords_new_list.csv",
                              show_col_types = FALSE))

# Attach unrecognized symbols
my_list <- c(" ","c'è","+"," ", my_word$stopwords, stopwords('italian'), stopwords("english"))

# Save my_list
#save(my_list,file="data/my_list.Rda")

doc.tokens <- tokens_select(doc.tokens, my_list, selection='remove')

DFM <- dfm(doc.tokens, tolower = TRUE)
```

9) Trim the data

Only words that occur in the top 20% of the distribution and in less than 30% of documents. Very frequent but document specific words.

```r
DFM_trimmed <- dfm_trim(DFM, min_termfreq = 0.80, termfreq_type = "quantile",
                        max_docfreq = 0.3, docfreq_type = "prop")

# Check the topfeatures
topfeatures(DFM_trimmed,15)
```

```
##    governo     grazie     lavoro      paese       anni presidente     grande
##      26036      20835      18314      16473      16317      14258      13656
##   italiani     italia   l'italia        via   politica  cittadini       bene
##      12011      11980      11752      11504       9964       9360       9311
##      forza
##       8505
```

10) Remove the emoji

```r
# Create a copy of the dfm
test <- DFM_trimmed
# Remove from the copy all the non ASCII carachters
test@Dimnames$features <- gsub("[^\x01-\x7F]", "", test@Dimnames$features)

# Check the difference from the list of features before and after apply gsub
a <- unique(test@Dimnames$features)
b <- unique(DFM_trimmed@Dimnames$features)
setdiff(b,a) #I have selected also words that cannot be removed

# Create an object with the features after remove non ASCII characters
c <- test@Dimnames$features
# Create an object with the original features
d <- DFM_trimmed@Dimnames$features

# Create the list of the removed features
diff <- setdiff(d,c)
emoji <- diff[diff %>% nchar() < 4]
emoji <- list(emoji)
emoji
# Now i can remove this list from the dfm
DFM_trimmed <- dfm_remove(DFM_trimmed, emoji)

#save(DFM_trimmed,file="data/dfm_trimmed.Rda")
```

11) Weight the feature frequencies in the dfm

```r
# Weight the frequency
dfm_weight <- DFM_trimmed %>%
  dfm_weight(scheme = "prop")
```

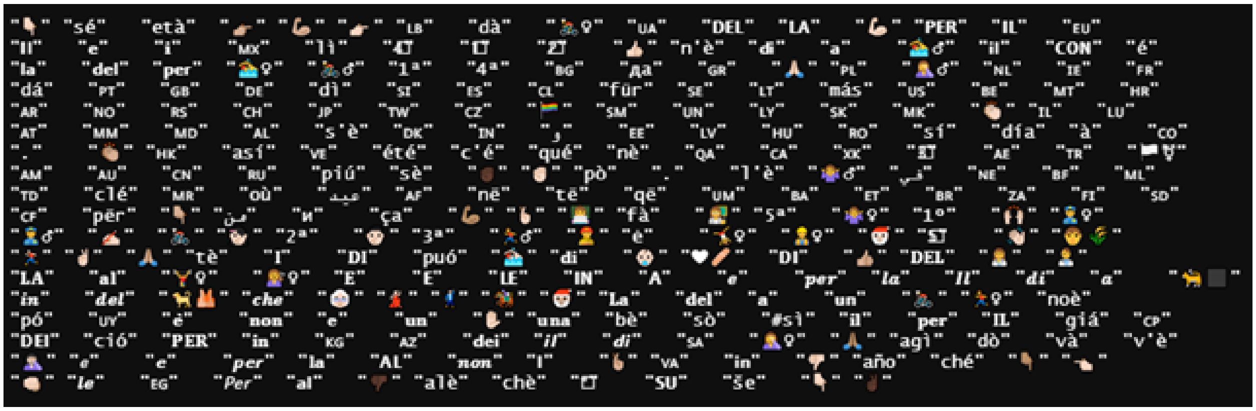


Figure 1: Emoji removed

Now the data are ready for the next analysis

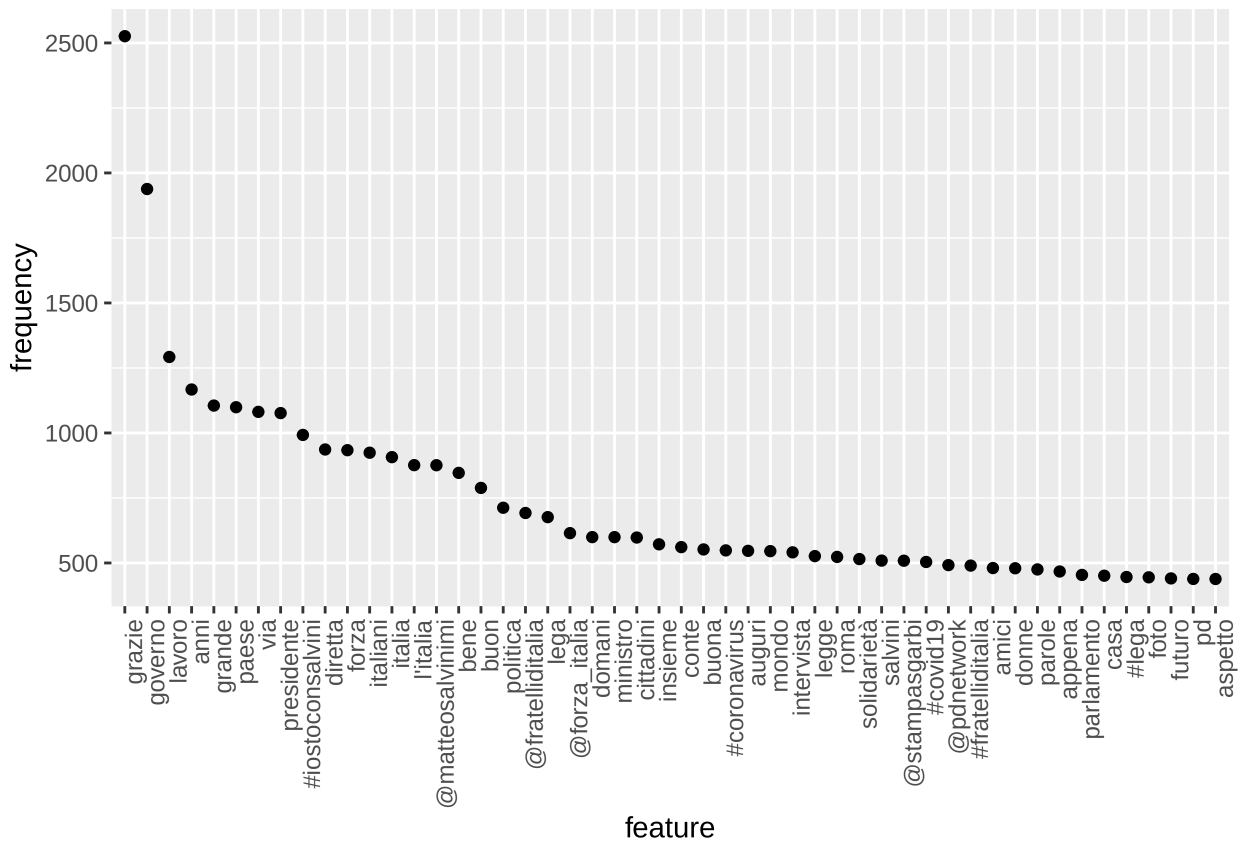
Part II: Preliminar analysis

1) Topfeatures frquency

```r
# Plot frequency of the topfeatures in the DFM
features_dfm <- textstat_frequency(dfm_weight, n = 50)

# Sort by reverse frequency order
features_dfm$feature <- with(features_dfm, reorder(feature, -frequency))

ggplot(features_dfm, aes(x = feature, y = frequency)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



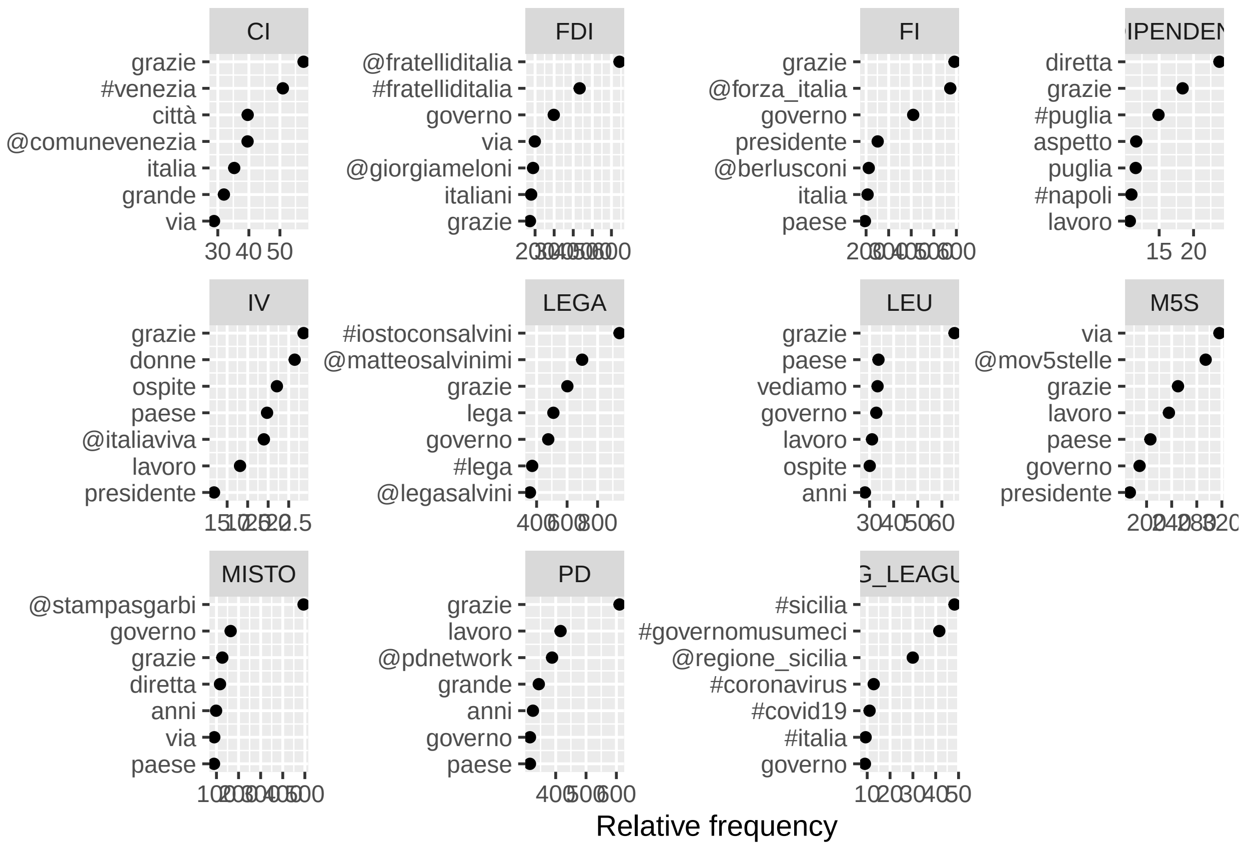
Relative frequency of the topfeatures by Party ID

```r
kable(unique(DFM_trimmed$party_id),col.names = "Party")
```

Party
PD
FDI
M5S
FI
REG_LEAGUES
MISTO
LEGA
IV
INDIPENDENTE
CI
LEU

```
# Plot relative frequency by party_id
freq_weight <- textstat_frequency(dfm_weight, n = 7,
                                  groups = dfm_weight$party_id)

ggplot(data = freq_weight, aes(x = nrow(freq_weight):1, y = frequency)) +
     geom_point() +
     facet_wrap(~ group, scales = "free") +
     coord_flip() +
     scale_x_continuous(breaks = nrow(freq_weight):1,
                        labels = freq_weight$feature) +
     labs(x = NULL, y = "Relative frequency")
```

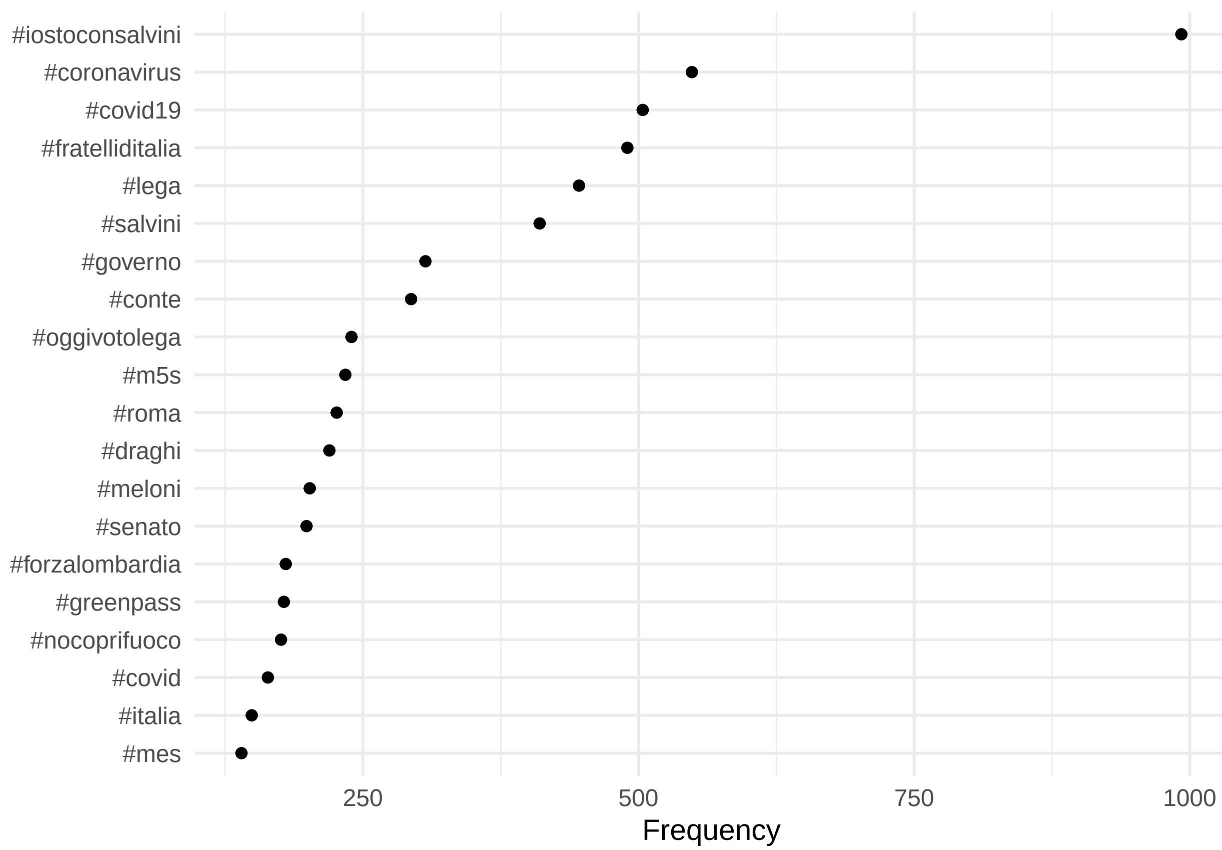


2) Most common hashtag

```
tag_dfm <- dfm_select(dfm_weight, pattern = "#*")
toptag <- names(topfeatures(tag_dfm, 20))
toptag
```

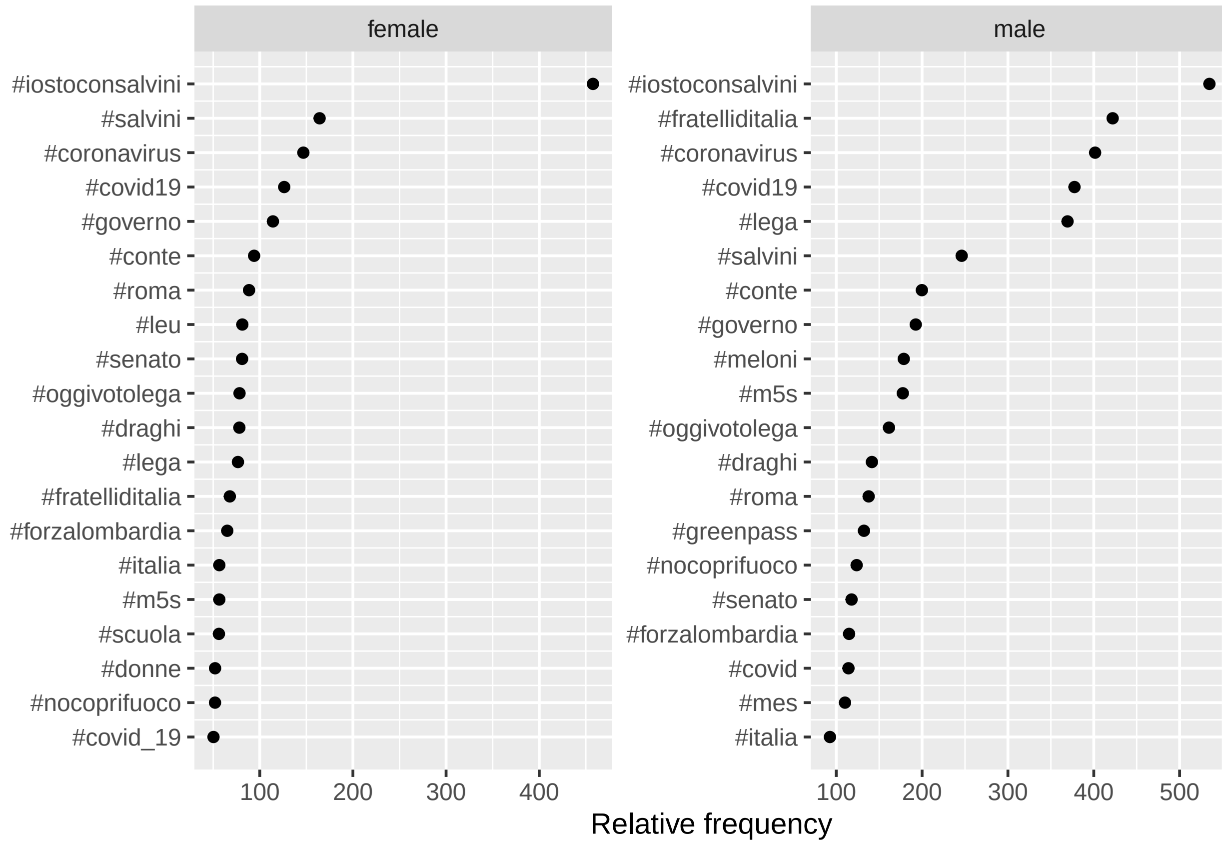
```
##  [1] "#iostoconsalvini" "#coronavirus"     "#covid19"         "#fratelliditalia"
##  [5] "#lega"            "#salvini"         "#governo"         "#conte"
##  [9] "#oggivotolega"    "#m5s"             "#roma"            "#draghi"
## [13] "#meloni"          "#senato"          "#forzalombardia"  "#greenpass"
## [17] "#nocoprifuoco"    "#covid"           "#italia"          "#mes"
```

```
tag_dfm %>%
  textstat_frequency(n = 20) %>%
  ggplot(aes(x = reorder(feature, frequency), y = frequency)) +
  geom_point() +
  coord_flip() +
  labs(x = NULL, y = "Frequency") +
  theme_minimal()
```



Most common hashtag by Gender

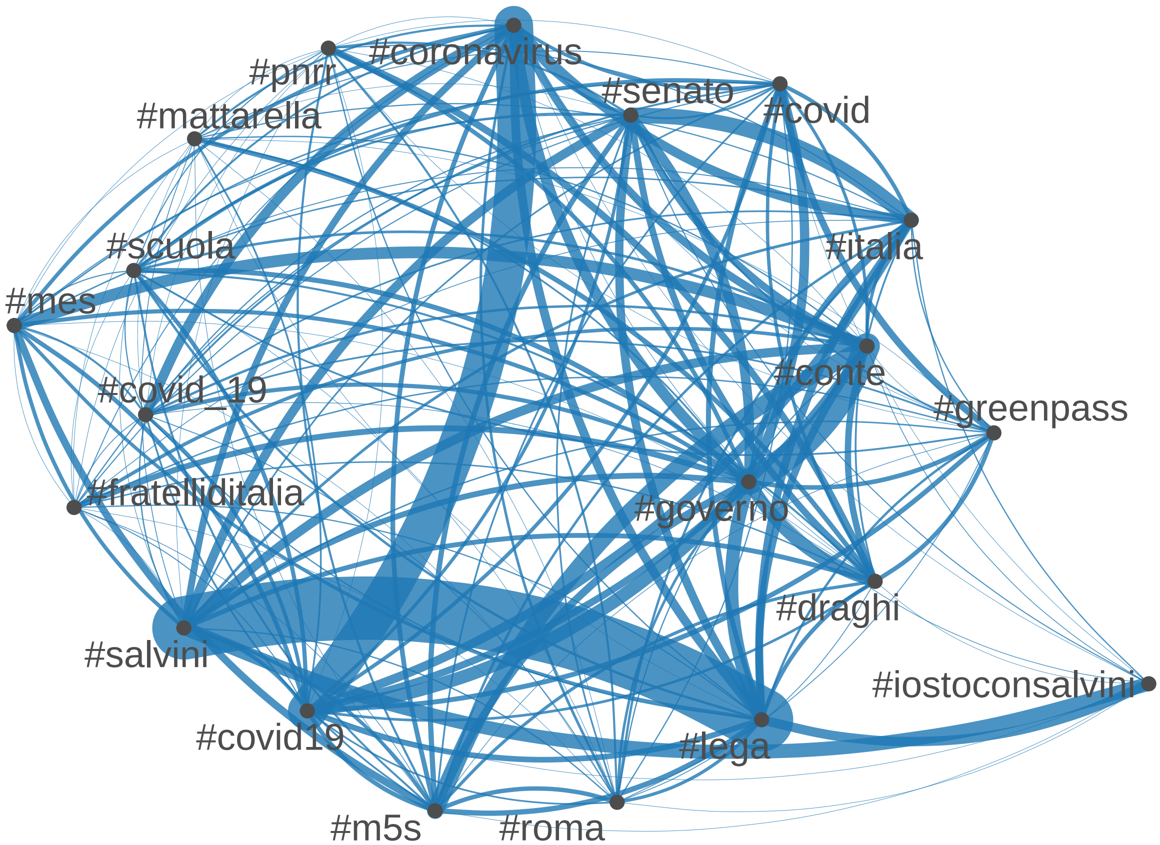
```r
tstat_freq <- textstat_frequency(tag_dfm, n = 20,
                                  groups = dfm_weight$genere)
```



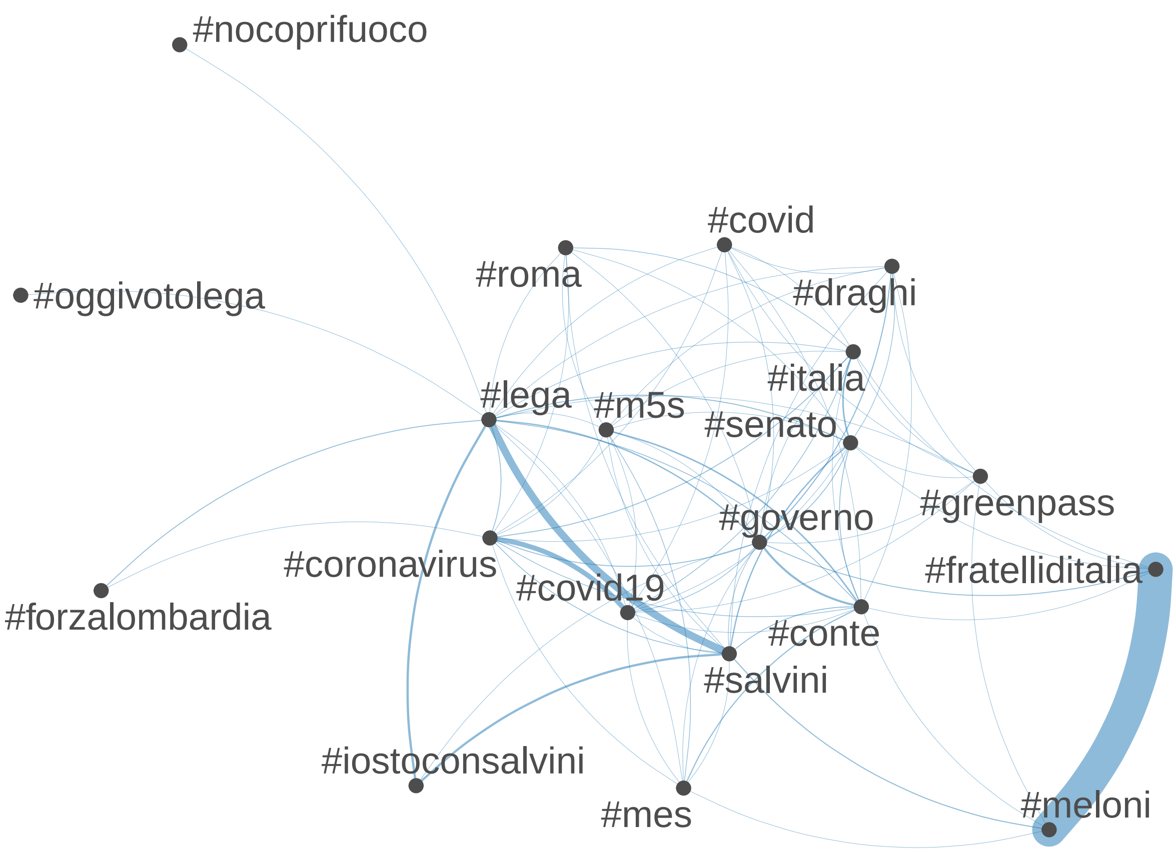
Co-occurrence Plot of hashtags

```r
# NOT WEIGHTED
tag_dfm_NOT_W <- dfm_select(DFM, pattern = "#*")
toptag_NOT <- names(topfeatures(tag_dfm_NOT_W, 20))

tag_fcm_NOT <- fcm(tag_dfm_NOT_W)
set.seed(666)
topgat_fcm_NOT <- fcm_select(tag_fcm_NOT, pattern = toptag_NOT)
textplot_network(topgat_fcm_NOT, min_freq = 0.1, edge_alpha = 0.8, edge_size = 5)
```




```
# WEIGHTED
tag_fcm <- fcm(tag_dfm)
set.seed(123)
topgat_fcm <- fcm_select(tag_fcm, pattern = toptag)
textplot_network(topgat_fcm)#, min_freq = 0.1, edge_alpha = 0.8, edge_size = 5)
```



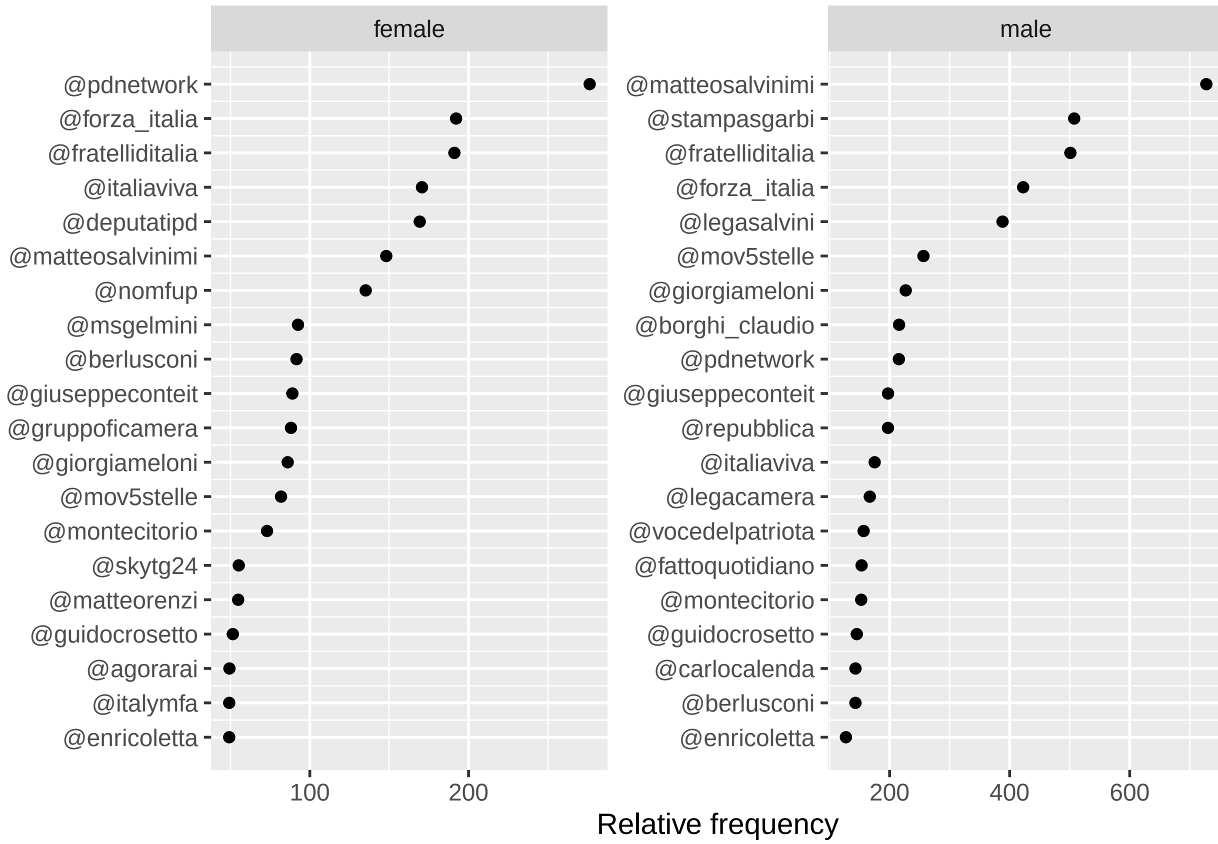
3) Most frequently mentioned usernames

```
user_dfm <- dfm_select(dfm_weight, pattern = "@*")
topuser <- names(topfeatures(user_dfm, 20, scheme = "docfreq"))
kable(topuser, col.names = "Most mentioned username")
```

Most mentioned username
@matteosalvinimi
@fratelliditalia
@forza_italia
@pdnetwork
@stampasgarbi
@mov5stelle
@legasalvini
@italiaviva
@giuseppeconteit
@giorgiameloni
@montecitorio
@deputatipd
@repubblica
@vocedelpatriota
@legacamera
@berlusconi
@matteorenzi
@fattoquotidiano
@enricoletta
@borghi_claudio

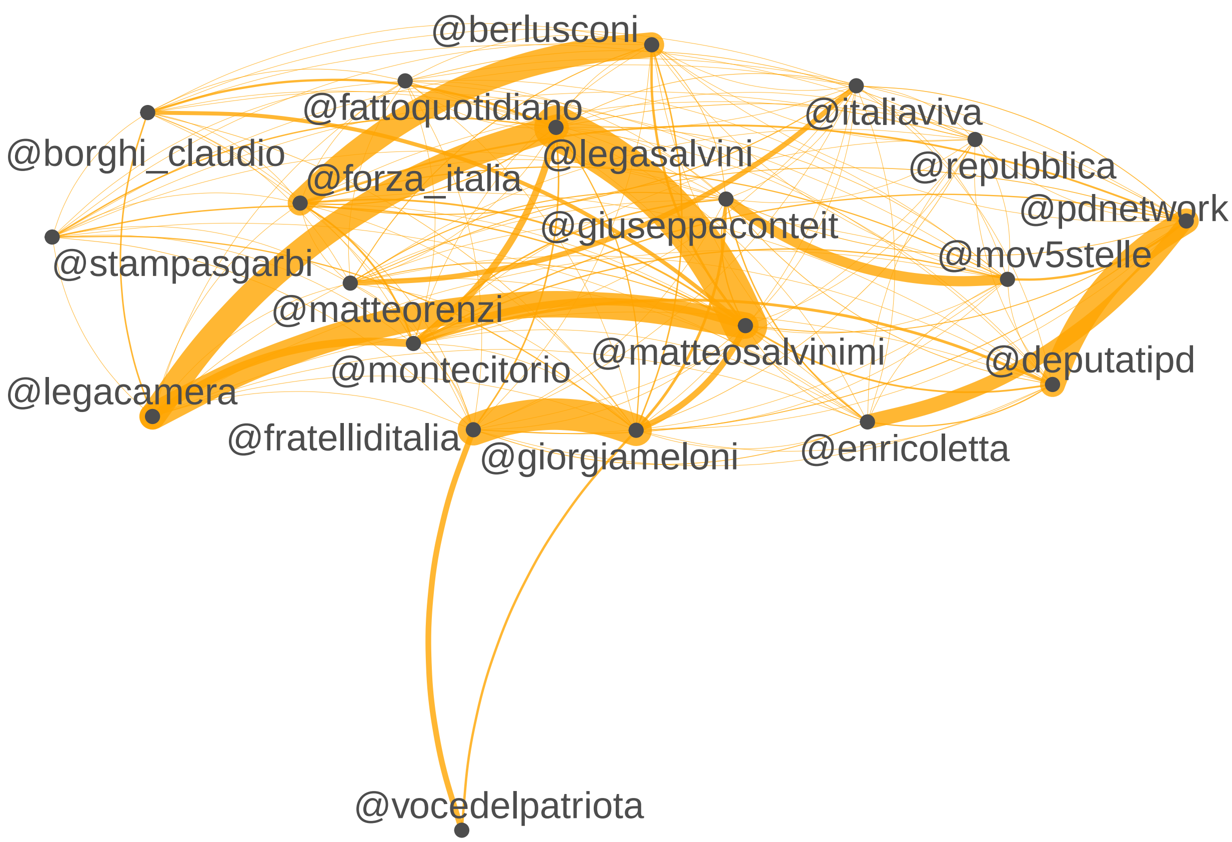
Most frequently mentioned usernames by gender

```
user_tstat_freq <- textstat_frequency(user_dfm, n = 20,
                                  groups = dfm_weight$genere)
```



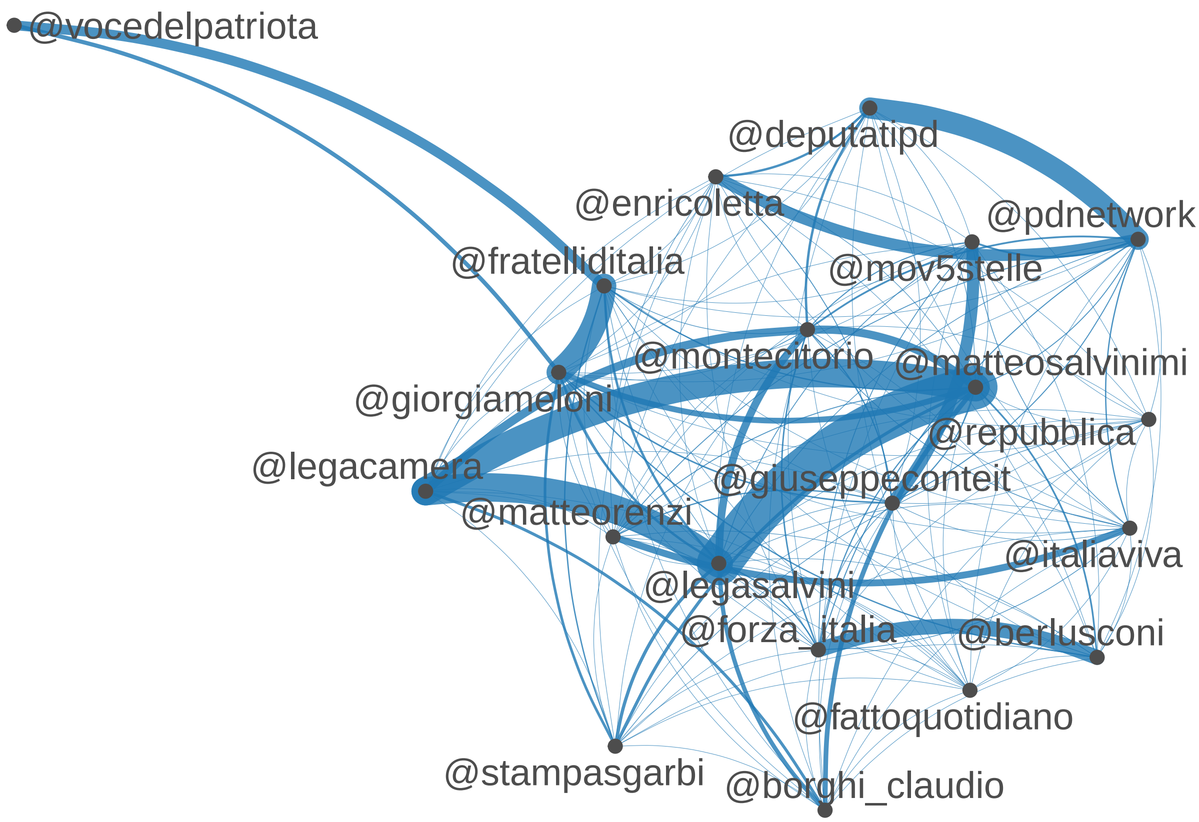
Co-occurrence plot of usernames

```
# WEIGHTED
user_fcm <- fcm(user_dfm)
set.seed(123)
user_fcm <- fcm_select(user_fcm, pattern = topuser)
textplot_network(user_fcm, min_freq = 0.1, edge_color = "orange", edge_alpha = 0.8, edge_size = 5)
```




```r
# NOT WEIGHTED
user_dfm_NOT_W <- dfm_select(DFM, pattern = "@*")
topuser_NOT <- names(topfeatures(user_dfm_NOT_W, 20, scheme = "docfreq"))

user_fcm_NOT <- fcm(user_dfm_NOT_W)
set.seed(6)
topuser_fcm_NOT <- fcm_select(user_fcm_NOT, pattern = topuser_NOT)
textplot_network(topuser_fcm_NOT, min_freq = 0.1, edge_alpha = 0.8, edge_size = 5)
```


@vocedelpatriota
@deputatipd
@enricoletta
@pdnetwork
@fratelliditalia
@mov5stelle
@montecitorio
@matteosalvinimi
@giorgiameloni
@repubblica
@legacamera
@giuseppeconteit
@matteorenzi
@italiaviva
@legasalvini
@forza_italia
@berlusconi
@fattoquotidiano
@stampasgarbi
@borghi_claudio