Preliminar analysis and recoding

Riccardo Ruta

5/2022

Contents

1) First import the dataset and check variables

```
# import the data
tw <-  read_csv("data/large_files/politicians_all_final_tweets.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

```
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"
```

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

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

Create the month variable

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

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

Check the number of month

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

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

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

Count the number of missing values

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

```
## [1] 153800
```

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	147306
party_id	0
genere	0
chamber	0
status	0
date	0
week	0
month	0

From that analysis i obtain 147306 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

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

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

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

5) Inspect that the variables correspond to the expectation

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

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

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

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

```
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] 32220 32221 32222 32223 32224
```

```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

```r
# 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

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

```
## [1] 390117
```

8) Create the DFM

```r
# 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'))

# 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)

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

```
##    governo     grazie     lavoro      paese       anni presidente     grande
##      25991      20760      18274      16444      16281      14215      13606
##   italiani     italia   l'italia        via   politica  cittadini       bene
##      11993      11955      11728      11495       9930       9331       9269
##      forza
##       8474
```

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")
```

Now the data are ready for the next analysis

10) Some preliminar analysis

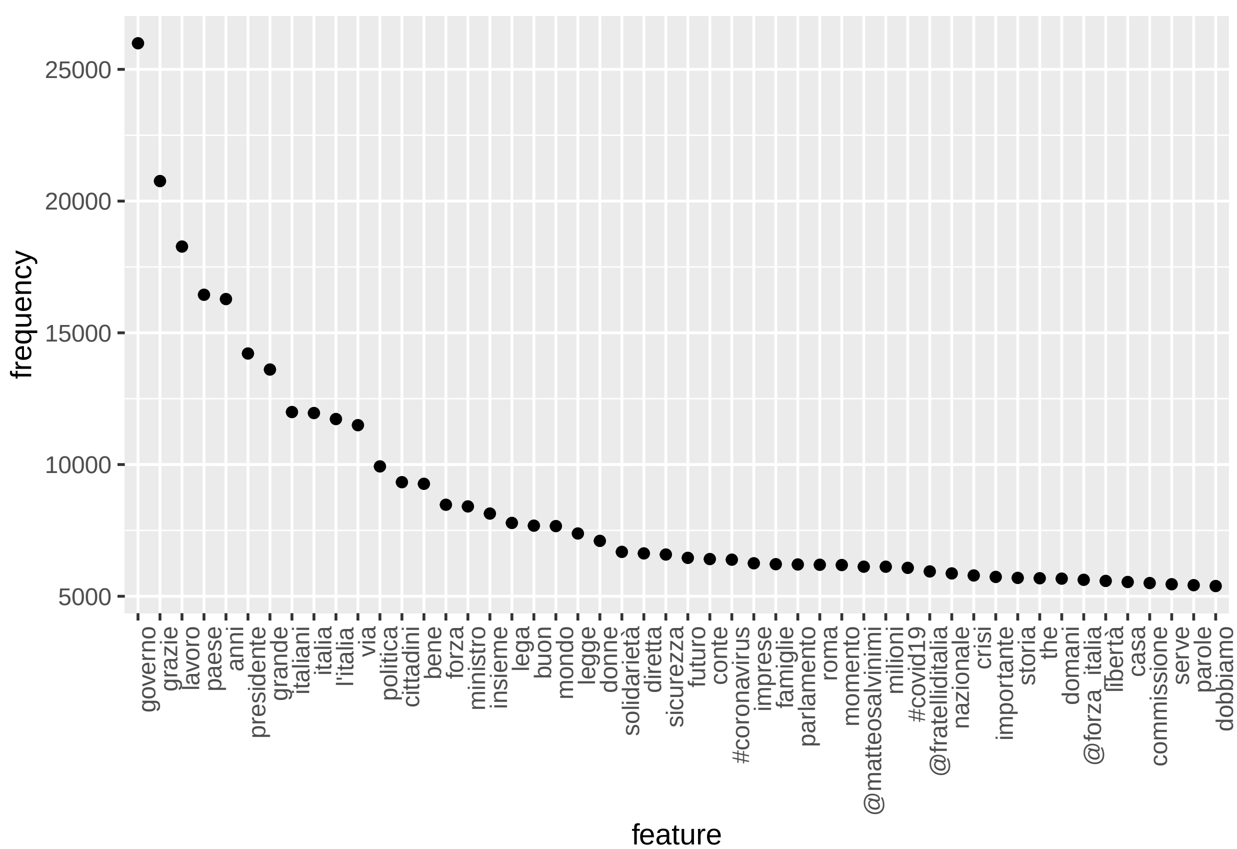
Topfeatures frquency

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

```
##      feature frequency rank docfreq group
## 1    governo     25991    1   24667   all
## 2     grazie     20760    2   19775   all
## 3     lavoro     18274    3   17107   all
## 4      paese     16444    4   16083   all
## 5       anni     16281    5   15420   all
## 6 presidente     14215    6   13444   all
```

```
# 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

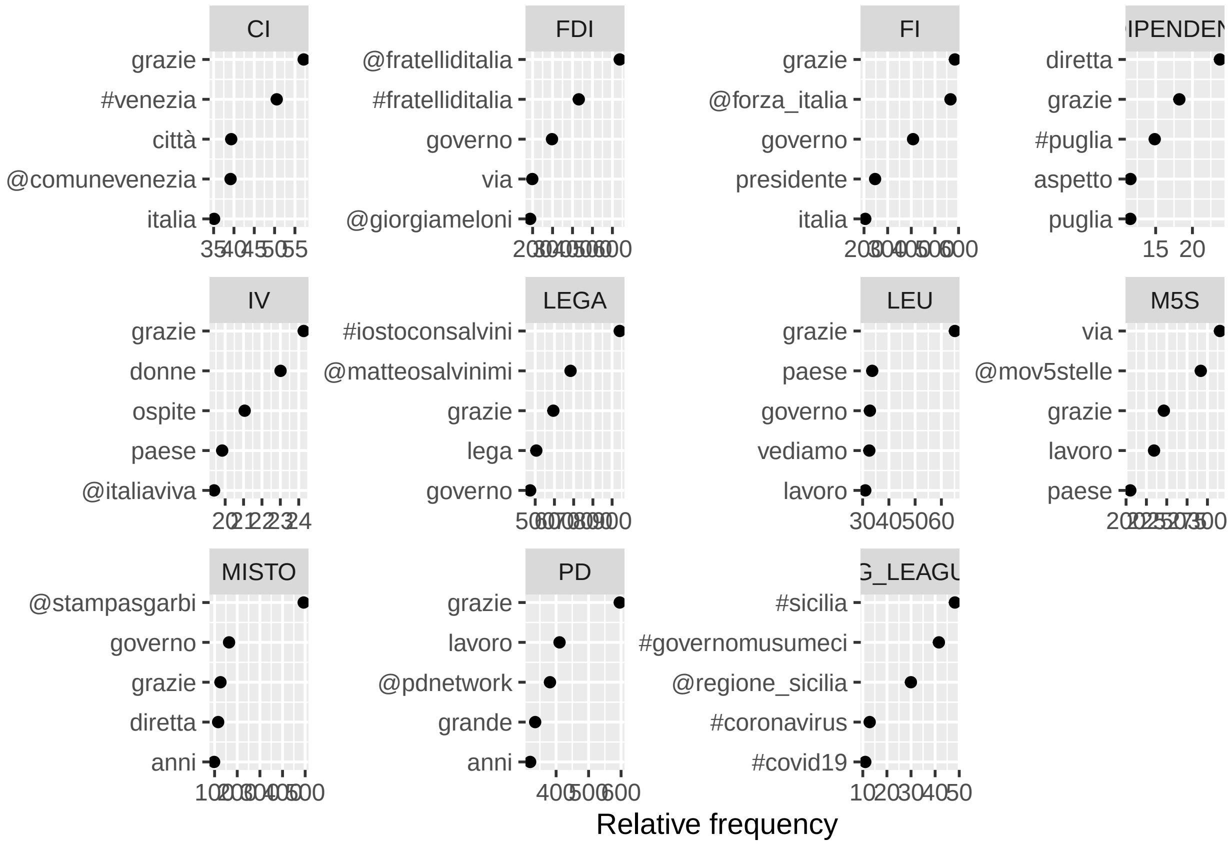
```
kable(unique(DFM_trimmed$party_id),col.names = "Party")
```

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

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

# Calculate relative frequency by president
freq_weight <- textstat_frequency(dfm_weight_pres, n = 5,
                                  groups = dfm_weight_pres$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")
```



Most frequent words by gender

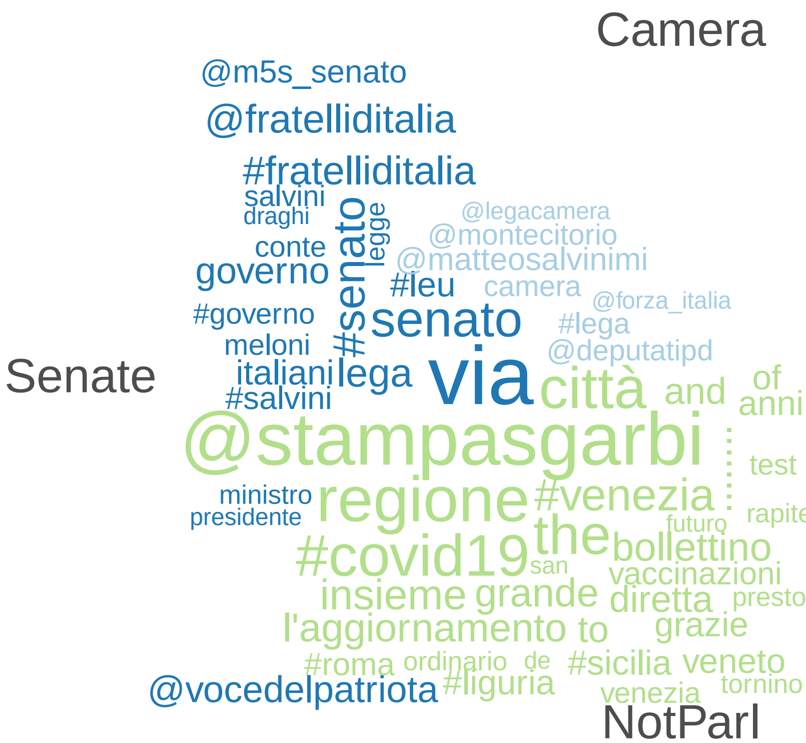
```
dfm_gender <- dfm_group(DFM_trimmed, groups = genere)

textplot_wordcloud(dfm_gender[c("male", "female"), ],
                   max_words = 80, comparison = TRUE)
```

Most frequent words by Chamber

```
dfm_chamber <- dfm_group(DFM_trimmed, groups = chamber)

textplot_wordcloud(dfm_chamber[c("Camera", "Senate", "NotParl"), ],
                   max_words = 80, comparison = TRUE)
```



Most common hashtag

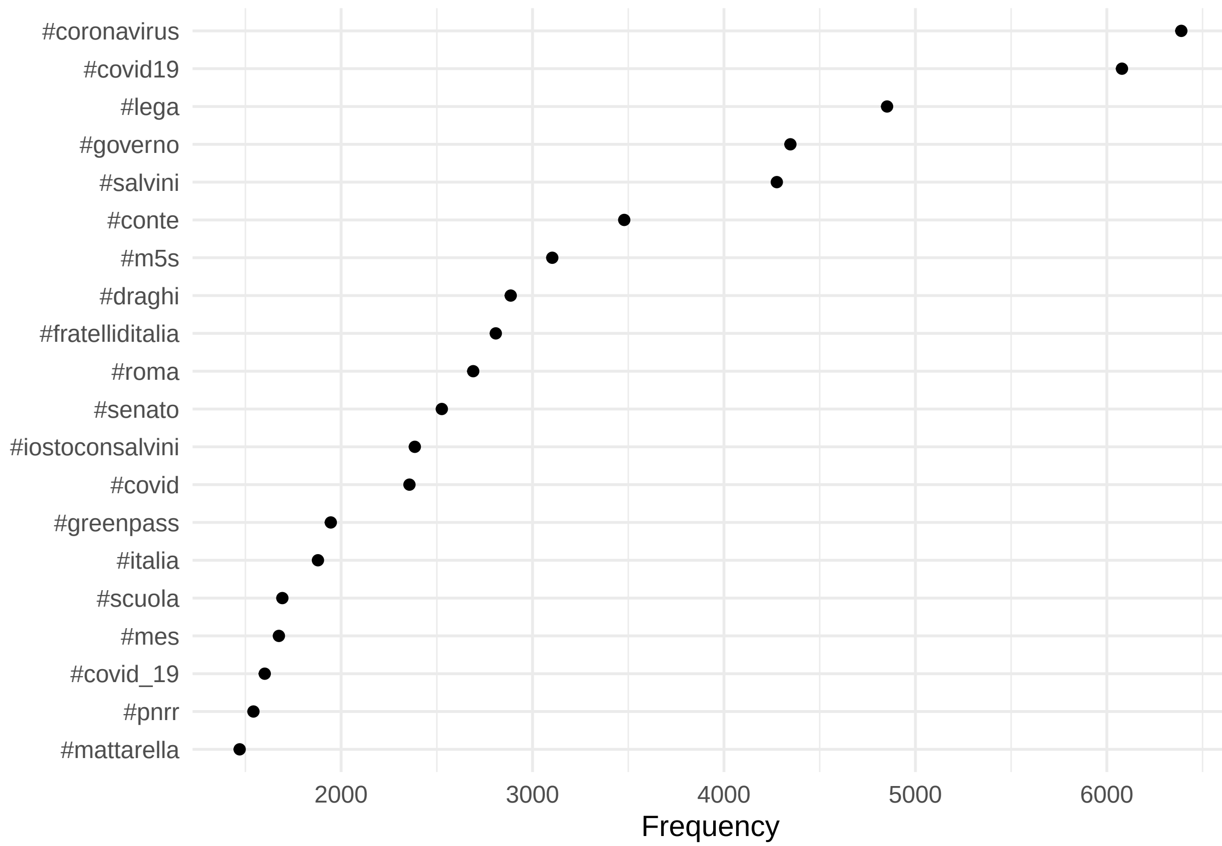
```r
tag_dfm <- dfm_select(DFM, pattern = "#*")
toptag <- names(topfeatures(tag_dfm, 20))
head(toptag)
```

```
## [1] "#coronavirus" "#covid19"     "#lega"        "#governo"     "#salvini"
## [6] "#conte"
```

```r
tstat_freq <- textstat_frequency(tag_dfm, n = 5, groups = genere)
head(tstat_freq, 20)
```

```
##         feature frequency rank docfreq  group
## 1  #coronavirus      1896    1    1893 female
## 2      #covid19      1870    2    1867 female
## 3      #governo      1697    3    1688 female
## 4      #salvini      1416    4    1411 female
## 5         #leu      1267    5    1260 female
## 6  #coronavirus      4493    1    4481   male
## 7      #covid19      4209    2    4202   male
## 8         #lega      4039    3    3953   male
## 9      #salvini      2860    4    2850   male
## 10     #governo      2650    5    2640   male
```

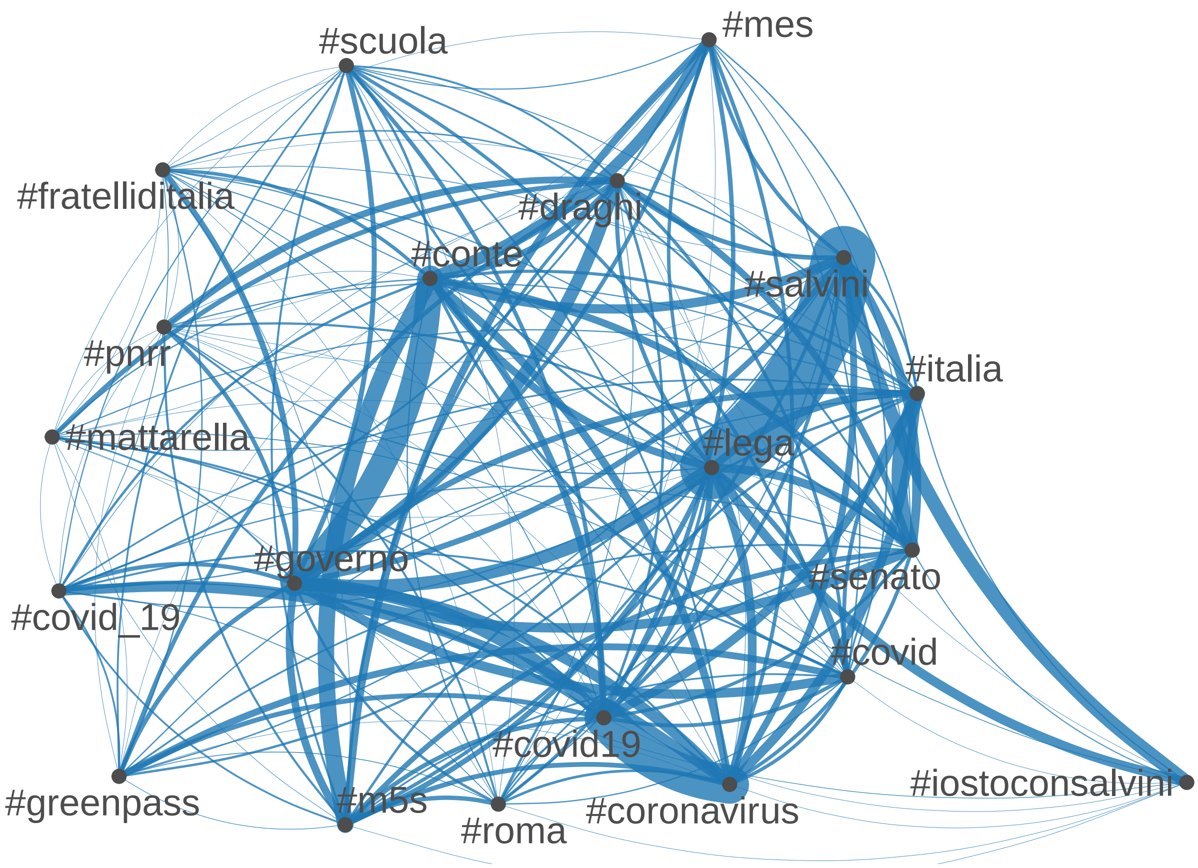
```r
tag_dfm %>%
  textstat_frequency(n = 20) %>%
  ggplot(aes(x = reorder(feature, frequency), y = frequency)) +
  geom_point() +
  coord_flip() +
  labs(x = NULL, y = "Frequency") +
  theme_minimal()
```



Co-occurrence Plot of hashtag

```r
tag_fcm <- fcm(tag_dfm)

topgat_fcm <- fcm_select(tag_fcm, pattern = toptag)
textplot_network(topgat_fcm, min_freq = 0.1, edge_alpha = 0.8, edge_size = 5)
```




```
ht <- str_extract_all(dataset$tweet_testo, '#[A-Za-z0-9_]+')
ht <- unlist(ht)
head(sort(table(ht), decreasing = TRUE))
```

Check most frequent hashtag extracted with regular expressions

```
## ht
##     #COVID19 #coronavirus        #Lega     #Salvini       #Conte      #Draghi
##         4467         4357         3790         3237         3224         2806
```

Extract most frequently mentioned usernames

```
user_dfm <- dfm_select(DFM, pattern = "@*")
topuser <- names(topfeatures(user_dfm, 20))
head(topuser)
```

```
## [1] "@matteosalvinimi" "@fratelliditalia" "@forza_italia"    "@pdnetwork"
## [5] "@stampasgarbi"    "@mov5stelle"
```

Feature-occurrence plot of usernames

```
user_fcm <- fcm(user_dfm)

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)
```