



UNIVERSITÀ DEGLI STUDI DI MILANO
FACOLTÀ DI SCIENZE POLITICHE,
ECONOMICHE E SOCIALI

**Political communication and populist rhetoric, an
analysis of Italian politicians in the digital arena.**

By

RICCARDO RUTA

DRAFT
DRAFT
DRAFT

07/22

Abstract

(the spacing is set to 1.5)

no more than 250 words for the abstract

- a description of the research question – what we know and what we don't know
- how the research has attempted to answer to this question
- a brief description of the methods
- brief results
- key conclusions that put the research into a larger context

Contents

1	Data cleaning	1
1.1	Import the dataset and check variables	1
1.2	Adjust date.time format	1
1.2.1	Check the conversion	2
1.3	Create the week variable	2
1.3.1	Check the variable	2
1.4	Create the month variable	3
1.4.1	Check the number of month	3
1.5	Count the number of missing values	3
1.5.1	Inspect where are the missings	4
1.5.2	Remove rows with missing tweets	5
1.6	Check that the variables make sense	6
1.6.1	Adjust the variable genere	6
1.6.2	Verify the substitution	7
1.7	Create a new dataset selecting only necessary informations	7
1.8	Create the corpus	8
1.9	Create the DFM	8
1.10	Trim the data	9
1.11	Remove the emoji	9
1.12	Take the proportion of the frequencies	11
1.12.1	Now the data are ready for the next analysis	11

2	Preliminar analysis	12
2.1	Topfeatures frquency	12
2.1.1	Relative frequency of the topfeatures by Party ID	13
2.2	Most common hashtag	15
2.2.1	Most common hashtag by Gender	16
2.2.2	Co-occurrence Plot of hashtags	17
2.3	Most frequently mentioned usernames	18
2.3.1	Most frequently mentioned usernames by gender	19
2.3.2	Co-occurrence plot of usernames	21
3	Dictionary analysis	23
3.1	Create the dictionary	23
3.2	Apply dictionary	25
3.3	Decadri_Boussalis_Grundl	26
3.3.1	Level of sparsity	26
3.4	Rooduijn_Pauwels_Italian	28
3.4.1	Level of sparsity	28
3.4.2	General level of populism in time	29
3.4.3	Most populist party	29
3.4.4	Most populist politician	30
3.5	Grundl_Italian_adapted	32
3.5.1	Level of sparsity	32
3.5.2	General level of populism in time	33

3.5.3	Most populist party	33
3.5.4	Most populist politician	34
3.6	Decadri_Boussalis	36
3.6.1	Level of sparsity	36
3.6.2	General level of populism in time	37
3.6.3	Most populist party	37
3.6.4	DISTRIBUTION OF PARTY POPULISM	38
3.6.5	Most populist politician	39
3.6.6	DISTRIBUTION OF POLITICIAN POPULISM	40
3.7	Compare the general level of populism in time for the dictionaries . .	41
4	Sentiment analysis	44
4.1	Inspect the dictionary	44
4.1.1	Clean text from dataframe	44
4.2	Create the filtered dataframes	45
4.3	Create nrc objects	46
4.4	Giorgia Meloni	48
4.4.1	Proportion of the emotion	48
4.4.2	Wordcloud of emotions	49
4.5	Giuseppe Conte	50
4.5.1	Proportion of the emotion	50
4.6	Matteo Renzi	52
4.6.1	Proportion of the emotion	52

4.6.2	Wordcloud of emotions	53
4.7	Matteo Salvini	54
4.7.1	Proportion of the emotion	54
4.7.2	Wordcloud of emotions	55
4.8	Enrico Letta	56
4.8.1	Proportion of the emotion	56
4.8.2	Wordcloud of emotions	57
4.9	Silvio Berlusconi	58
4.9.1	Proportion of the emotion	58
4.9.2	Wordcloud of emotions	59
4.10	Roberto Speranza	60
4.10.1	Proportion of the emotion	60
4.10.2	Wordcloud of emotions	61
5	LDA Topic model analysis	62
5.1	CREATE THE DTM	62
5.1.1	Remove all the account's mentions	62
5.2	FIND THE BEST NUMBER OF TOPICS K	62
5.2.1	Search the best number of Topics comparing coherence and exclusivity values	62
5.2.2	Plot the values of coherence and exclusivity in order to find the best K	63
5.3	ANALISYS OF THE TOPICS	64
5.3.1	Repeat the analysis selecting k = 11	64

5.3.2	The most important terms from the model, for each topic . .	64
6	FER: Facial Emotion Recognition Analysis	67
6.1	Report on the analysis made with FER Puthon package	67

1 Data cleaning

1.1 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

1.2 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))
```


1.2.1 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

1.3 Create the week variable

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

1.3.1 Check the variable

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

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

```
## [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
```

1.4 Create the month variable

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

1.4.1 Check the number of month

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

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

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

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

1.5 Count the number of missing values

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

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

1.5.1 Inspect where are the missings

```
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.

1.5.2 Remove rows with missing tweets

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

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

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

1.6 Check that the variables make sense

```
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"
```

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

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

1.6.1 Adjust the variable genere

```
# Remove space from genere variable [RUN ONLY ONCE!]
```

```
a <- unique(tw$genere)
```

```
a[3]
```

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

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

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

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

1.6.2 Verify the substitution

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

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

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

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

Now all the variables are ready for next steps

1.7 Create a new dataset selecting only necessary informations

```
# Select variables for the analysis
```

```
dataset <- tw %>% select(nome, tweet_testo, genere, party_id, chamber, status, date, w  
colnames(dataset)
```

```
## [1] "nome" "tweet_testo" "genere" "party_id" "chamber"
```

```
## [6] "status" "date" "week" "month"
```

1.8 Create the corpus

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

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

1.9 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(" "))  
  
# 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)
```

1.10 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.

```
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
##	italiani	italia	l'italia	via	politica	cittadini
##	12011	11980	11752	11504	9964	9360
##	forza					
##	8505					

1.11 Remove the emoji

```
# Create a copy of the dfm
```

```
test <- DFM_trimmed
```

```
# Remove from the copy all the non ASCII caracters
```

```
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")

```

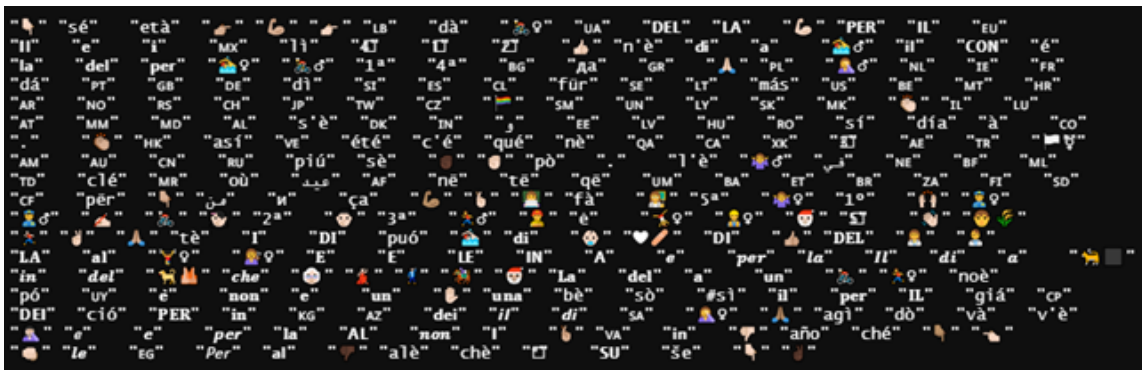


Figure 1: Emoji removed

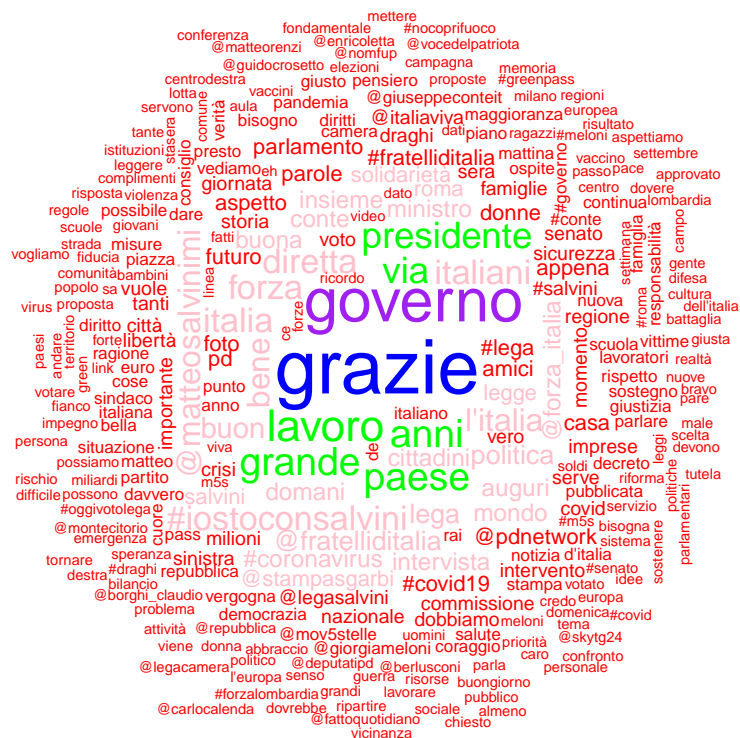
1.12 Take the proportion of the frequencies

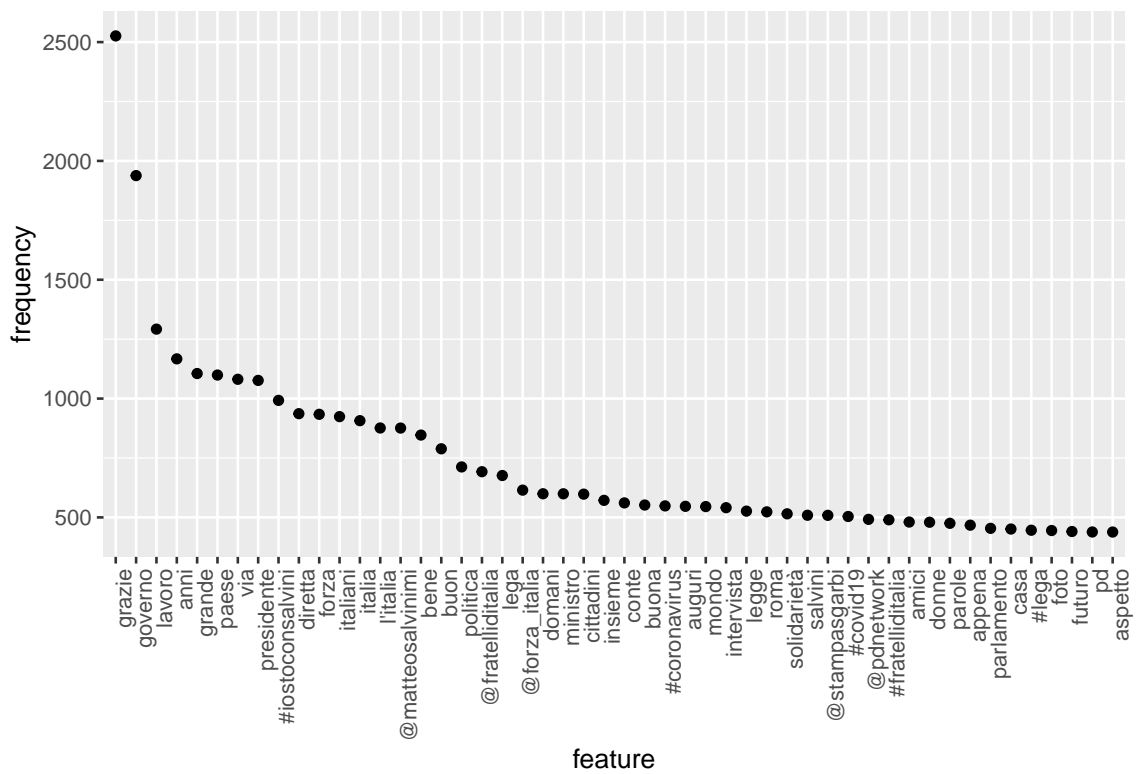
```
# Weight the frequency  
dfm_weight <- DFM_trimmed %>%  
  dfm_weight(scheme = "prop")
```

1.12.1 Now the data are ready for the next analysis

2 Preliminar analysis

2.1 Topfeatures frquency

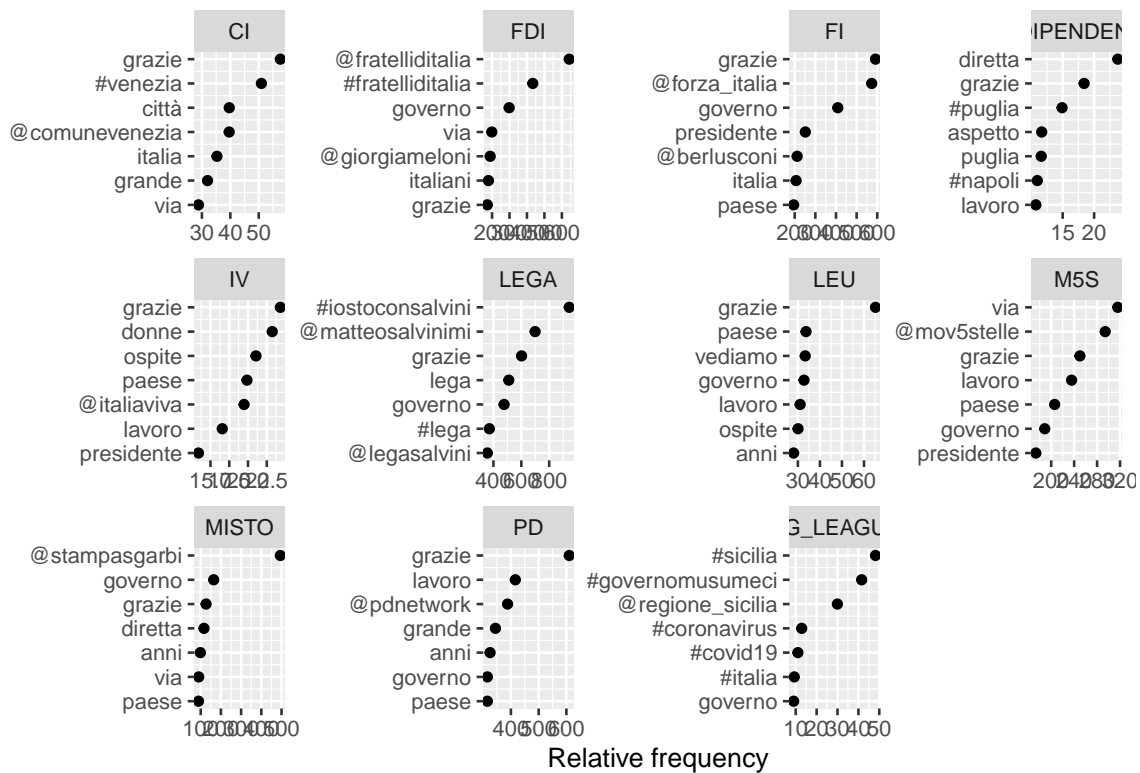




2.1.1 Relative frequency of the topfeatures by Party ID

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

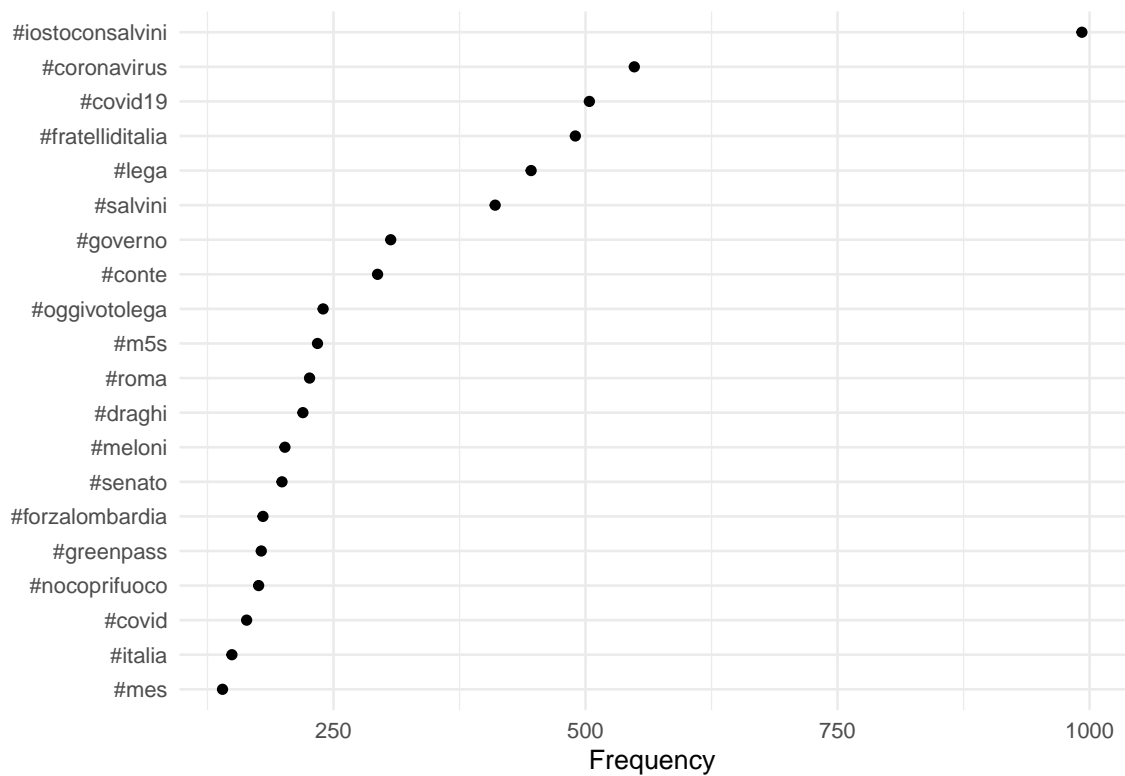
Party
PD
FDI
M5S
FI
REG_LEAGUES
MISTO
LEGA
IV
INDIPENDENTE
CI
LEU



2.2 Most common hashtag

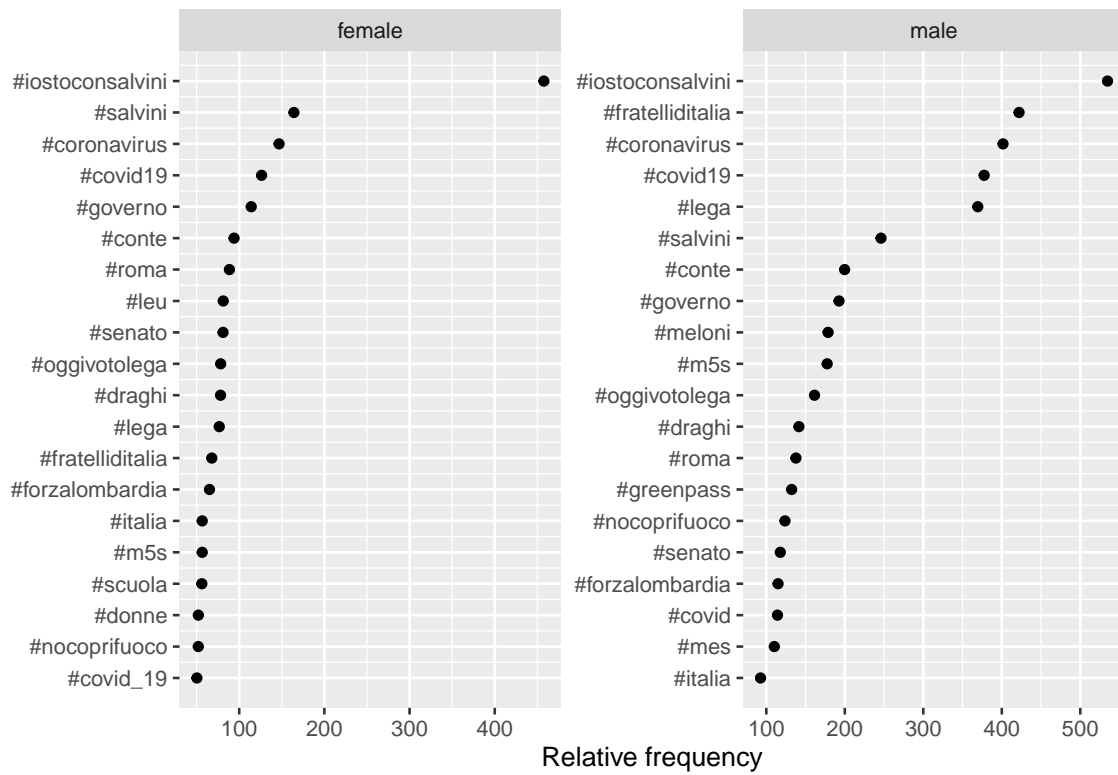
```
tag_dfm <- dfm_select(dfm_weight, pattern = "#*")
toptag <- names(topfeatures(tag_dfm, 20))
toptag
```

```
## [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"
```



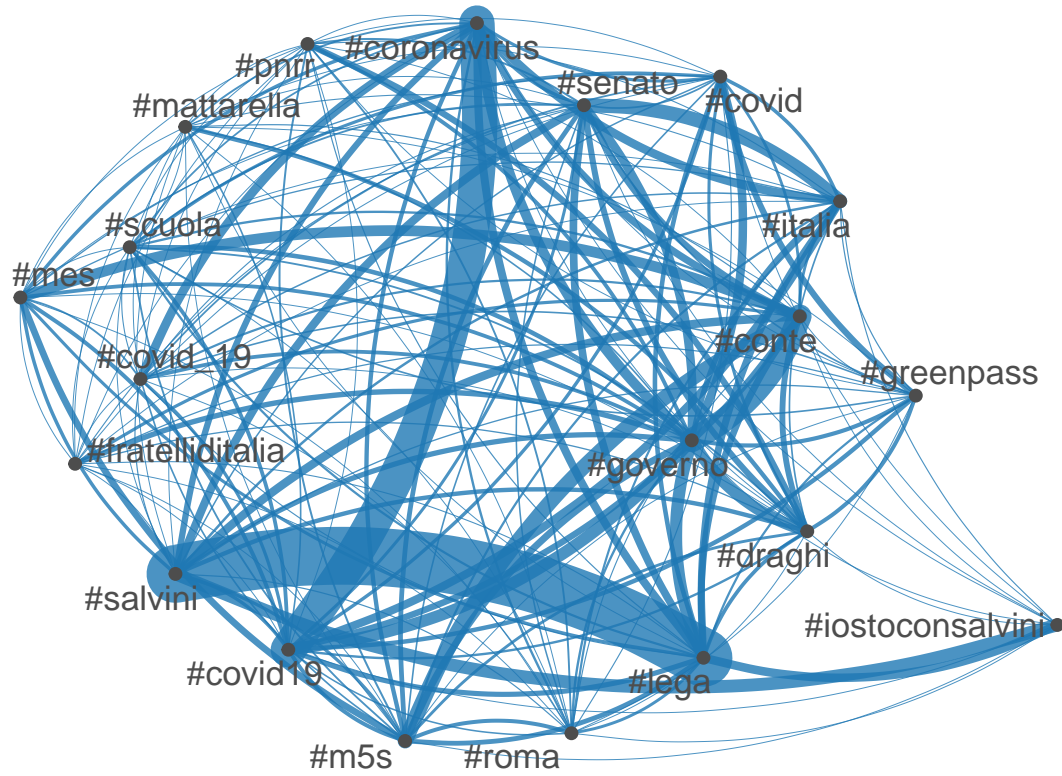
2.2.1 Most common hashtag by Gender

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

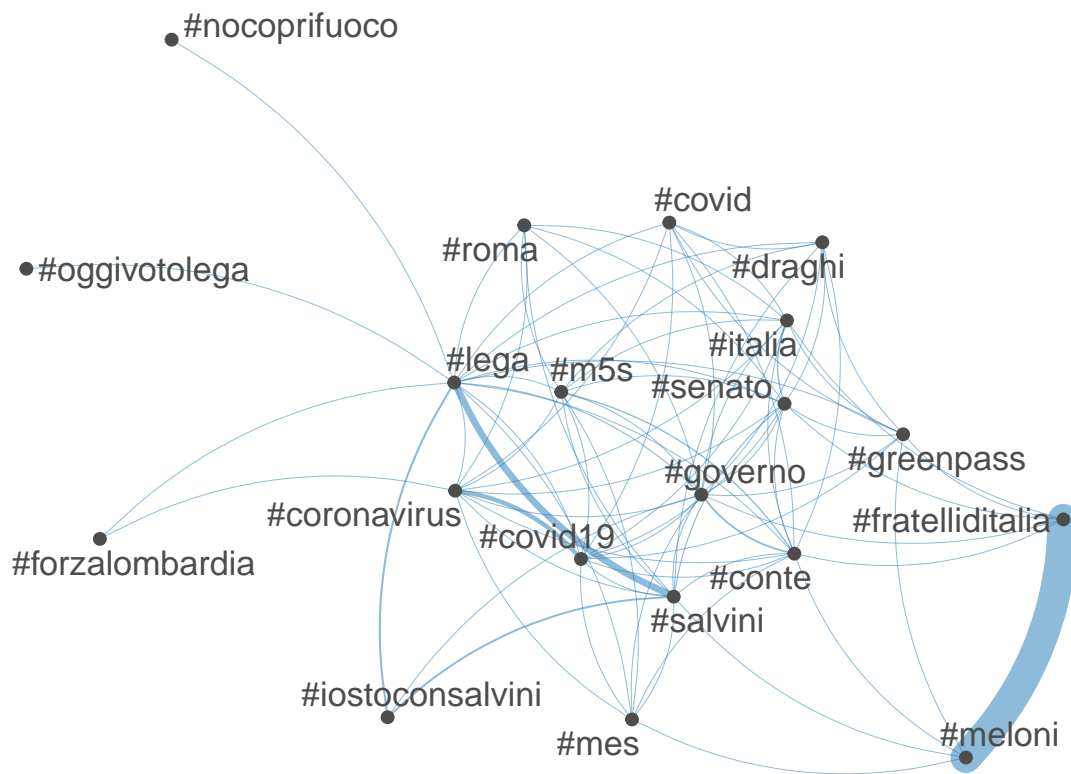


2.2.2 Co-occurrence Plot of hashtags

Not weighted



Weighted



2.3 Most frequently mentioned usernames

```

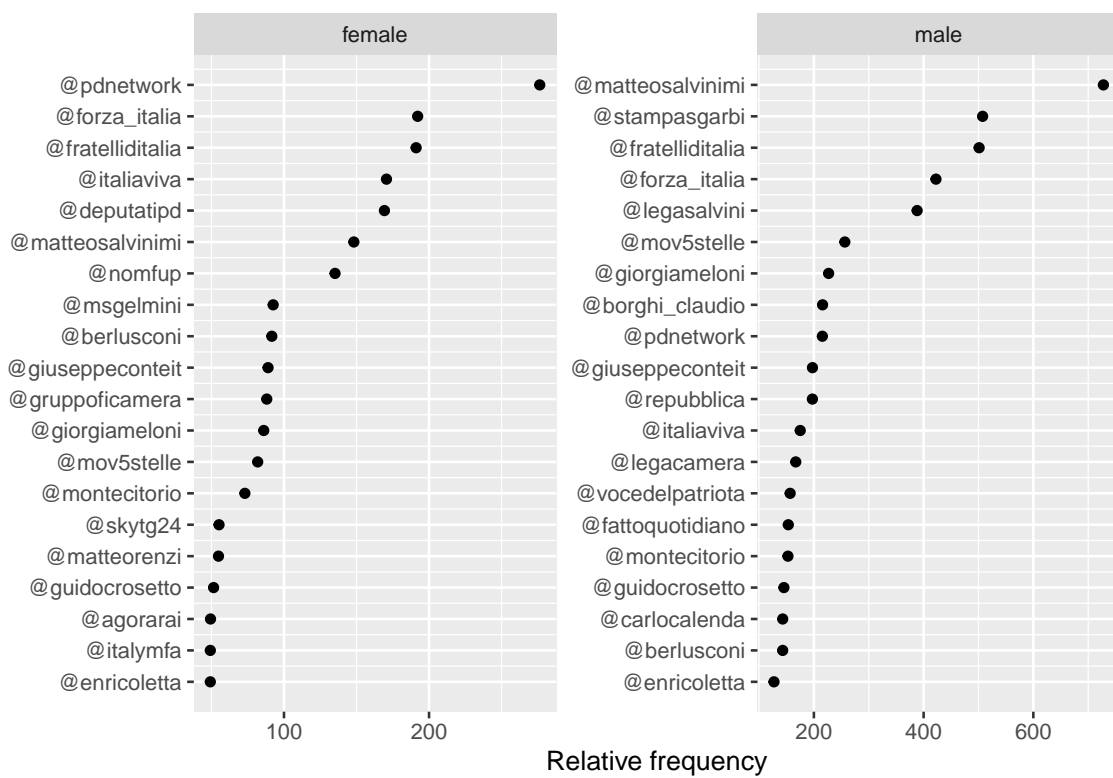
user_dfm <- dfm_select(dfm_weight, pattern = "@*")
topuser <- names(topfeatures(user_dfm, 20, scheme = "docfreq"))
kable(topuser, col.names = "Most mentioned username")

```

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

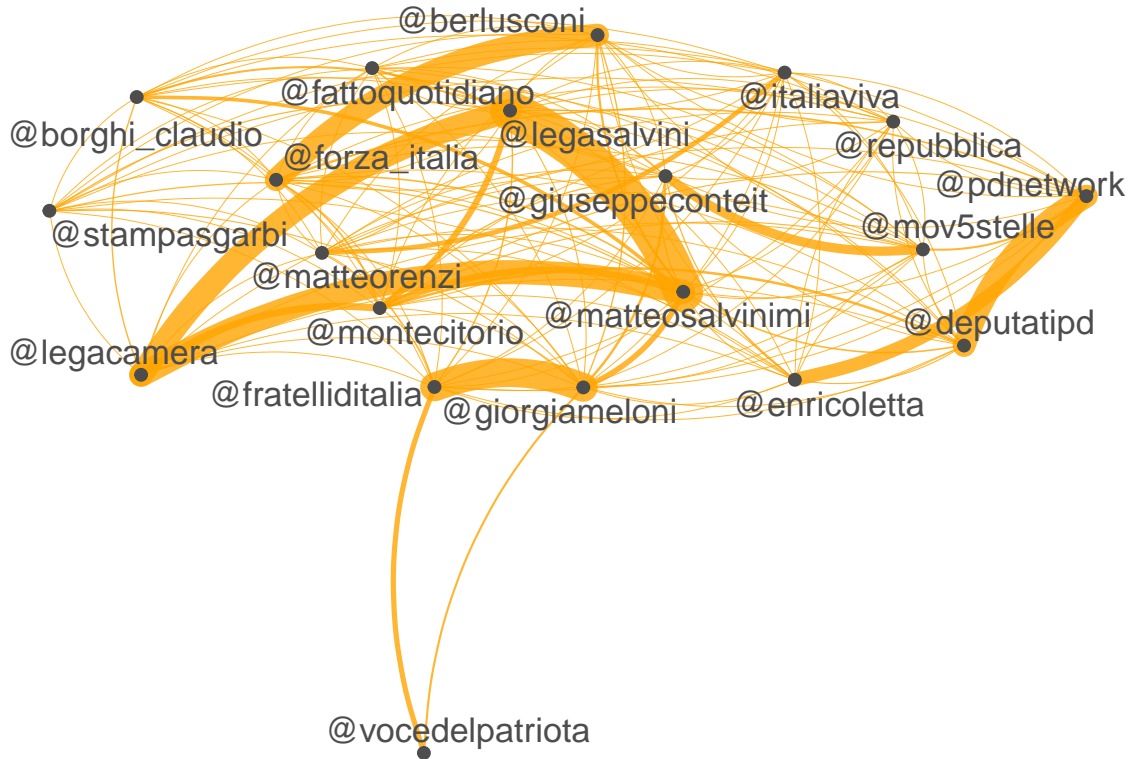
2.3.1 Most frequently mentioned usernames by gender

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



2.3.2 Co-occurrence plot of usernames

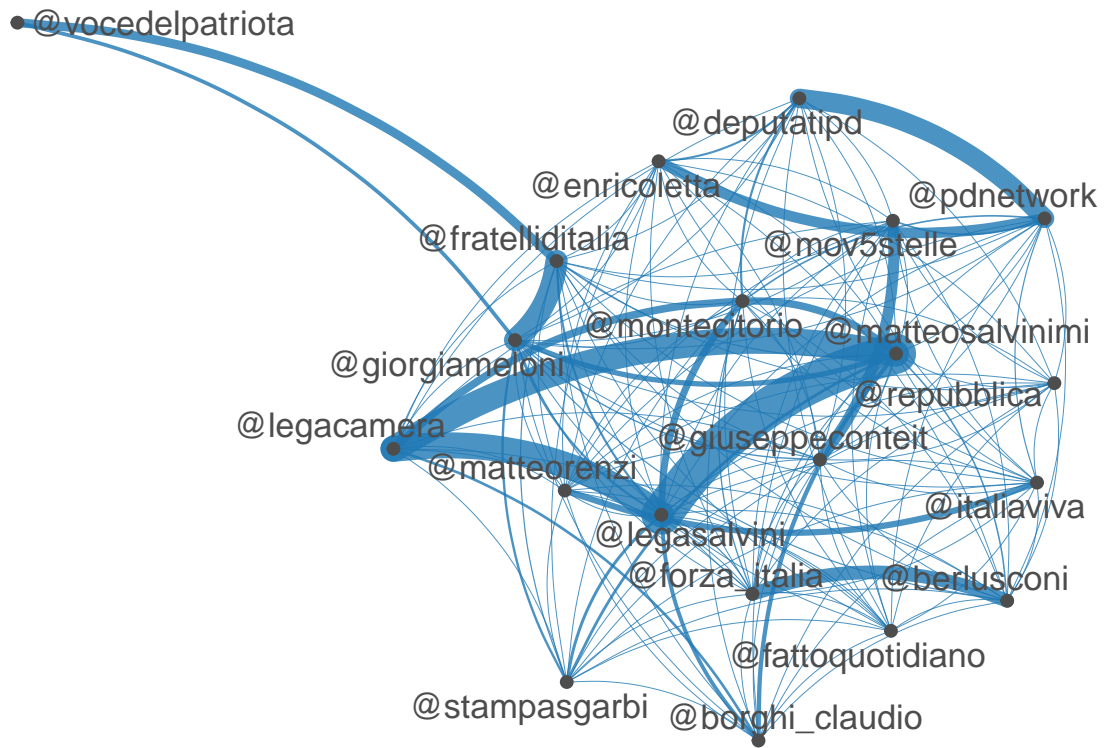
Weighted



Not weighted

```
# 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)
```



3 Dictionary analysis

At the level of political parties, which ones make most use of populist rhetoric?

At the level of individual politicians, which ones make most use of populist rhetoric?

I use 3 dictionary to perform the analysis

- Rooduijn & Pauwels: Rooduijn, M., and T. Pauwels. 2011. “Measuring Populism: Comparing Two Methods of Content Analysis.” *West European Politics* 34 (6): 1272–1283.
- Decadri & Boussalis: Decadri, S., & Boussalis, C. (2020). Populism, party membership, and language complexity in the Italian chamber of deputies. *Journal of Elections, Public Opinion and Parties*, 30(4), 484-503.
- Grundl: Gründl J. Populist ideas on social media: A dictionary-based measurement of populist communication. *New Media & Society*. December 2020.
- Decadri & Boussalis + Grundl: this is simply a more extended version of the D&B dictionary, which also contains some terms taken from Grundl.

3.1 Create the dictionary

```
# import dictionaries file
dict <- read_excel("data/populism_dictionaries.xlsx")
variable.names(dict)

## [1] "Rooduijn_Pauwels_Italian"
## [2] "Grundl_Italian_adapted"
```

```
## [3] "Decadri_Boussalis"
## [4] "Decadri_Boussalis_Grundl_People"
## [5] "Decadri_Boussalis_Grundl_Common Will"
## [6] "Decadri_Boussalis_Grundl_Elite"
```

create the dictionary

```
Rooduijn_Pauwels_Italian <-
  dictionary(list(populism =
                  (dict$Rooduijn_Pauwels_Italian
                   [!is.na(dict$Rooduijn_Pauwels_Italian)])))

Grundl_Italian_adapted <-
  dictionary(list(populism =
                  dict$Grundl_Italian_adapted
                  [!is.na(dict$Grundl_Italian_adapted)]))

Decadri_Boussalis <-
  dictionary(list(populism =
                  dict$Decadri_Boussalis
                  [!is.na(dict$Decadri_Boussalis)]))

Decadri_Boussalis_Grundl <-
  dictionary(list(people =
                  dict$Decadri_Boussalis_Grundl_People
                  [!is.na(dict$Decadri_Boussalis_Grundl_People)],
                  common_will =
                  dict$`Decadri_Boussalis_Grundl_Common Will`
                  [!is.na(dict$`Decadri_Boussalis_Grundl_Common Will`)],
                  elite =
```

```
dict$Decadri_Boussalis_Grundl_Elite  
[!is.na(dict$Decadri_Boussalis_Grundl_Elite)])
```

3.2 Apply dictionary

3.3 Decadri_Boussalis_Grundl

3.3.1 Level of sparsity

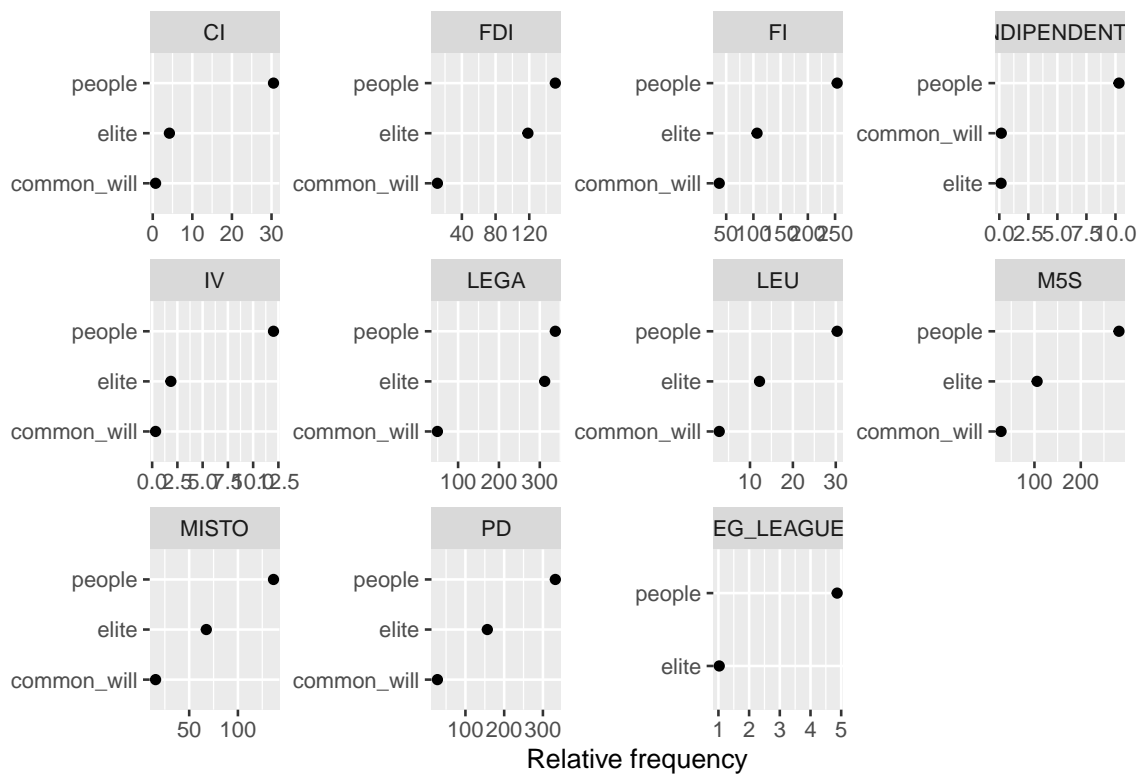
daily: 12.08%

weekly: 0.55%

monthly: 0%

```
# Daily Dictionary analysis with Decadri_Boussalis_Grundl on the whole dataset
dfm_dict1 <- dfm_lookup(dfm_weight, dictionary = Decadri_Boussalis_Grundl)
# Group by date
dfm_by_date1 <- dfm_group(dfm_dict1, groups= date)
#dfm_by_date1
# Group by week
dfm_by_week1 <- dfm_group(dfm_dict1, groups= week)
#dfm_by_week1
# Group by month
dfm_by_month1 <- dfm_group(dfm_dict1, groups= month)

#kable(dfm_by_month1)
```



Looking at the populist rhetoric for each party divided into the 3 components people-centrism, anti-elitism and common-will, we note that the most frequent components is People-centrism.

3.4 Rooduijn_Pauwels_Italian

3.4.1 Level of sparsity

daily: 0.60%

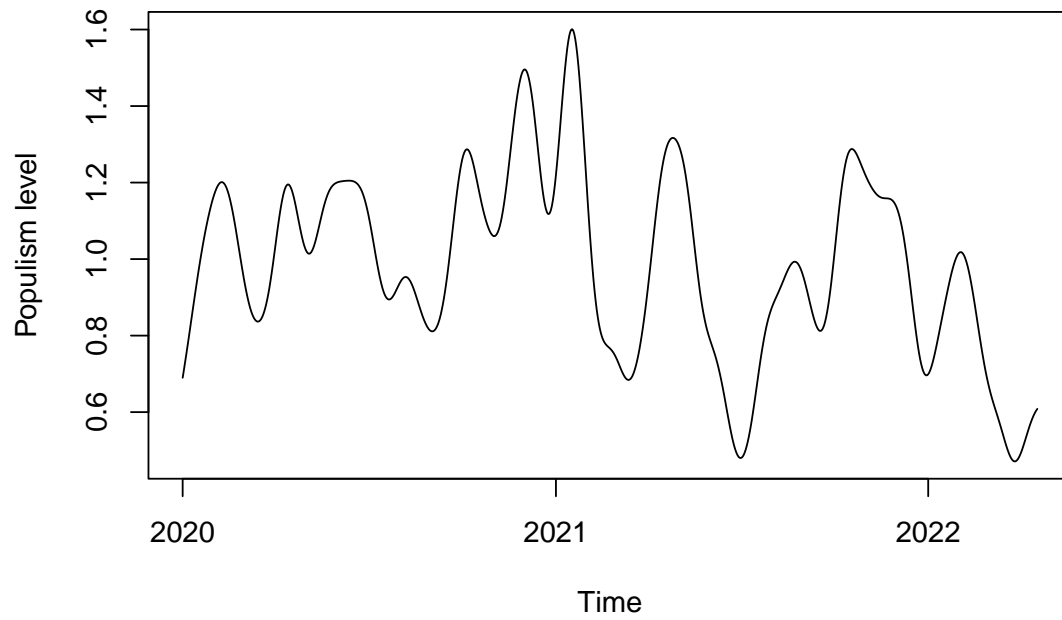
weekly: 0.0%

monthly: 0.0%

```
# Daily Dictionary analysis with Rooduijn_Pauwels_Italian on the whole dataset
dfm_dict2 <- dfm_lookup(dfm_weight, dictionary = Rooduijn_Pauwels_Italian)
# Group by date
dfm_by_date2 <- dfm_group(dfm_dict2, groups= date)
#dfm_by_date2
# Group by week
dfm_by_week2 <- dfm_group(dfm_dict2, groups= week)
#dfm_by_week2
# Group by month
dfm_by_month2 <- dfm_group(dfm_dict2, groups= month)

#kable(dfm_by_month2)
```

3.4.2 General level of populism in time



3.4.3 Most populist party

```
# Most populist party
dfm_dict2_tstat_party <- textstat_frequency(dfm_dict2, groups = party_id)
kable(dfm_dict2_tstat_party %>% slice_max(frequency, n = 20))
```

	feature	frequency	rank	docfreq	group
6	populism	303.9474786	1	1919	LEGA
10	populism	149.7512641	1	1671	PD
2	populism	113.7388243	1	1124	FDI
3	populism	98.6906136	1	941	FI
8	populism	87.6625041	1	1119	M5S
9	populism	60.9720255	1	669	MISTO
7	populism	11.7023384	1	175	LEU
1	populism	3.7116701	1	45	CI
5	populism	1.8540424	1	26	IV
11	populism	1.0264294	1	11	REG_LEAGUES
4	populism	0.0833333	1	1	INDIPENDENTE

3.4.4 Most populist politician

```
dict2_tstat_nome <- textstat_frequency(dfm_dict2, groups = nome)

kable(dict2_tstat_nome %>% slice_max(frequency, n = 20))
```

	feature	frequency	rank	docfreq	group
194	populism	42.115152	1	146	FERRERO Roberta
472	populism	15.910436	1	160	SGARBI Vittorio
341	populism	14.112659	1	77	MORANI Alessia
24	populism	13.999694	1	52	BALDELLI Simone
179	populism	13.821584	1	48	FAGGI Antonella
271	populism	13.095709	1	149	LANNUTTI Elio
217	populism	12.884799	1	39	FREGOLENT Sonia
450	populism	12.806346	1	64	RUSPANDINI Massimo
326	populism	12.518396	1	192	MELONI Giorgia
427	populism	12.257891	1	40	RIVOLTA Erica
106	populism	10.788399	1	68	CECCHETTI Fabrizio
283	populism	10.783981	1	108	LOLLOBRIGIDA Francesco
260	populism	10.778644	1	76	IEZZI Igor Giancarlo
230	populism	10.648954	1	155	GARNERO SANTANCHE' Daniela
303	populism	10.133849	1	78	MALAN Lucio
447	populism	9.885108	1	29	RUFA Gianfranco
455	populism	9.561830	1	93	SALVINI Matteo
360	populism	9.110910	1	105	NOBILI Luciano
35	populism	8.689617	1	57	BAZZARO Alex
501	populism	8.495460	1	32	TONELLI Gianni

3.5 Grundl_Italian_adapted

3.5.1 Level of sparsity

daily: 0.24%

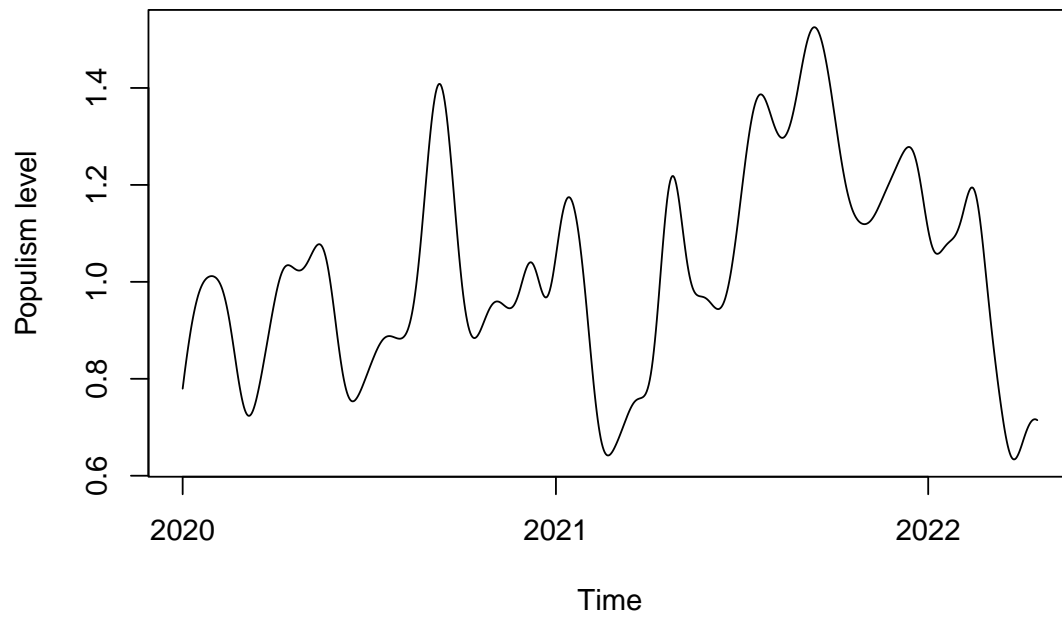
weekly: 0.0%

monthly: 0%

```
# Daily Dictionary analysis with Grundl_Italian_adapted on the whole dataset
dfm_dict3 <- dfm_lookup(dfm_weight, dictionary = Grundl_Italian_adapted)
# Group by date
dfm_by_date3<- dfm_group(dfm_dict3, groups= date)
#dfm_by_date3
# Group by week
dfm_by_week3 <- dfm_group(dfm_dict3, groups= week)
#dfm_by_week3
# Group by month
dfm_by_month3 <- dfm_group(dfm_dict3, groups= month)

#kable(dfm_by_month3)
```

3.5.2 General level of populism in time



3.5.3 Most populist party

```
# Most populist party  
dict_3_tstat_party <- textstat_frequency(dfm_dict3, groups = party_id)  
kable(dict_3_tstat_party %>% slice_max(frequency, n = 20))
```


	feature	frequency	rank	docfreq	group
6	populism	225.678708	1	2075	LEGA
10	populism	153.269683	1	2017	PD
8	populism	133.053746	1	1724	M5S
3	populism	131.838292	1	1524	FI
2	populism	99.425177	1	1087	FDI
9	populism	86.092041	1	997	MISTO
7	populism	15.213765	1	231	LEU
1	populism	10.602522	1	157	CI
5	populism	2.559005	1	40	IV
4	populism	1.983671	1	31	INDIPENDENTE
11	populism	1.505044	1	22	REG_LEAGUES

3.5.4 Most populist politician

```
dict_3_tstat_nome <- textstat_frequency(dfm_dict3, groups = nome)

kable(dict_3_tstat_nome %>% slice_max(frequency, n = 20))
```

	feature	frequency	rank	docfreq	group
287	populism	23.033031	1	240	LANNUTTI Elio
210	populism	19.501980	1	110	FERRERO Roberta
562	populism	19.042283	1	131	VITO Elio
275	populism	16.483870	1	120	IEZZI Igor Giancarlo
494	populism	15.974269	1	184	SGARBI Vittorio
341	populism	11.063928	1	159	MELONI Giorgia
15	populism	10.731212	1	120	ANZALDI Michele
298	populism	10.659433	1	98	LOLLOBRIGIDA Francesco
74	populism	10.645964	1	97	BORGHI Claudio
476	populism	9.238862	1	122	SALVINI Matteo
248	populism	9.004085	1	139	GARNERO SANTANCHE' Daniela
96	populism	8.438949	1	103	CANGINI Andrea
546	populism	8.339166	1	106	URSO Adolfo
224	populism	8.162373	1	101	FONTANA Lorenzo
472	populism	7.850014	1	68	RUSPANDINI Massimo
44	populism	7.832168	1	120	BERGESIO Giorgio Maria
165	populism	7.565932	1	92	DE MARTINI Guido
141	populism	7.036558	1	43	CROSETTO Guido
446	populism	7.000320	1	47	RIVOLTA Erica
359	populism	6.861311	1	73	MORELLI Alessandro

3.6 Decadri_Boussalis

3.6.1 Level of sparsity

daily: 0%

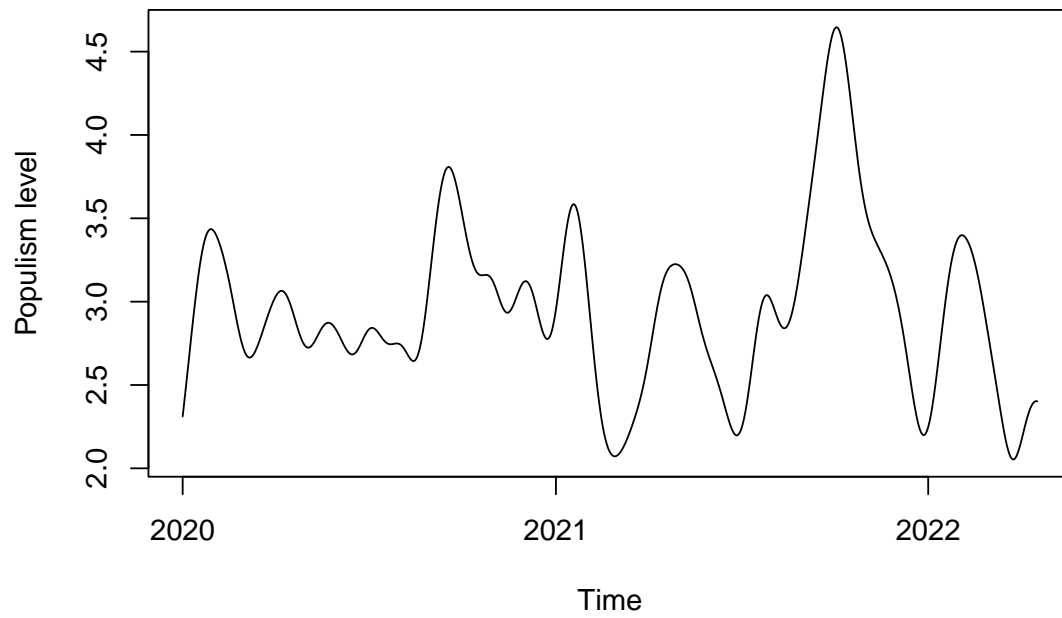
weekly: 0.0%

monthly: 0%

```
# Daily Dictionary analysis with Decadri_Boussalis on the whole dataset
dfm_dict4 <- dfm_lookup(dfm_weight, dictionary = Decadri_Boussalis)
# Group by date
dfm_by_date4<- dfm_group(dfm_dict4, groups= date)
#dfm_by_date4
# Group by week
dfm_by_week4 <- dfm_group(dfm_dict4, groups= week)
#dfm_by_week4
# Group by month
dfm_by_month4 <- dfm_group(dfm_dict4, groups= month)

#kable(dfm_by_month4)
```

3.6.2 General level of populism in time



3.6.3 Most populist party

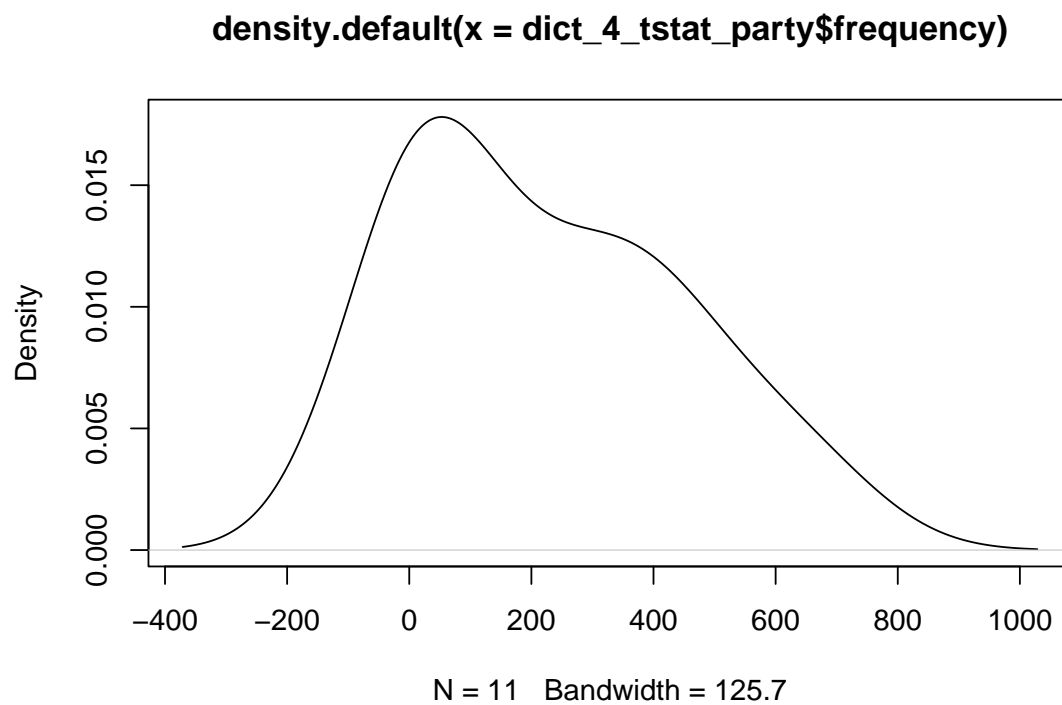
```
# Most populist party  
dict_4_tstat_party <- textstat_frequency(dfm_dict4, groups = party_id)  
kable(dict_4_tstat_party %>% slice_max(frequency, n = 20))
```

	feature	frequency	rank	docfreq	group
6	populism	651.348390	1	5672	LEGA
10	populism	493.532735	1	6417	PD
8	populism	376.966170	1	5178	M5S
3	populism	376.609606	1	4532	FI
2	populism	270.814483	1	2960	FDI
9	populism	202.466904	1	2463	MISTO
7	populism	44.919508	1	659	LEU
1	populism	35.105322	1	506	CI
5	populism	14.132863	1	197	IV
4	populism	10.615825	1	153	INDIPENDENTE
11	populism	6.122696	1	93	REG_LEAGUES

3.6.4 DISTRIBUTION OF PARTY POPULISM

```
x <- dict_4_tstat_party$frequency

d <- density(dict_4_tstat_party$frequency)
d$y <- d$y * length(dict_4_tstat_party$frequency)
plot(d)
```



3.6.5 Most populist politician

```
dict_4_tstat_nome <- textstat_frequency(dfm_dict4, groups = nome)

kable(dict_4_tstat_nome %>% slice_max(frequency, n = 20))
```

	feature	frequency	rank	docfreq	group
236	populism	62.66405	1	282	FERRERO Roberta
560	populism	41.70723	1	443	SGARBI Vittorio
329	populism	34.85565	1	397	LANNUTTI Elio
391	populism	33.15912	1	496	MELONI Giorgia
344	populism	32.36912	1	358	LOLLOBRIGIDA Francesco
540	populism	29.61242	1	368	SALVINI Matteo
27	populism	27.44810	1	135	BALDELLI Simone
280	populism	26.74696	1	372	GARNERO SANTANCHE' Daniela
530	populism	24.85093	1	184	ROTONDI Gianfranco
68	populism	24.50676	1	252	BONACCINI Stefano
220	populism	24.35617	1	122	FAGGI Antonella
317	populism	24.31241	1	207	IEZZI Igor Giancarlo
128	populism	23.82148	1	195	CECCHETTI Fabrizio
585	populism	23.63509	1	327	TAJANI Antonio
80	populism	22.82617	1	240	BORGHI Claudio
161	populism	21.54784	1	158	CROSETTO Guido
39	populism	21.35229	1	202	BAZZARO Alex
47	populism	21.29380	1	318	BERGESIO Giorgio Maria
535	populism	20.92822	1	140	RUSPANDINI Massimo
365	populism	20.38171	1	185	MALAN Lucio

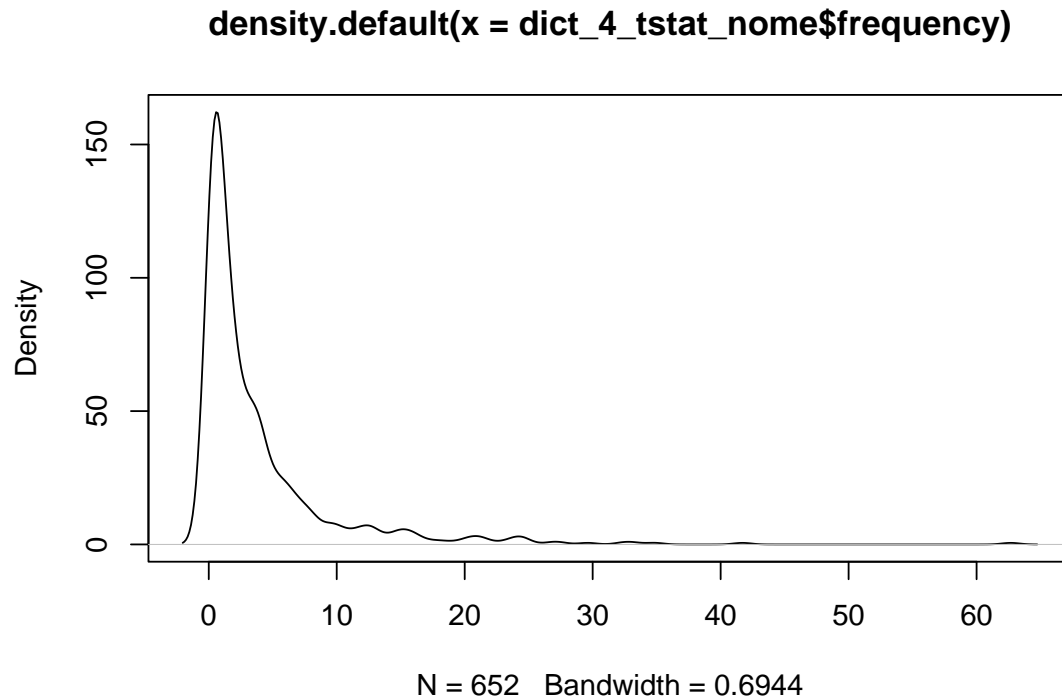
3.6.6 DISTRIBUTION OF POLITICIAN POPULISM

```
x <- dict_4_tstat_nome$frequency

d <- density(dict_4_tstat_nome$frequency)

d$y <- d$y * length(dict_4_tstat_nome$frequency)
```

```
plot(d)
```

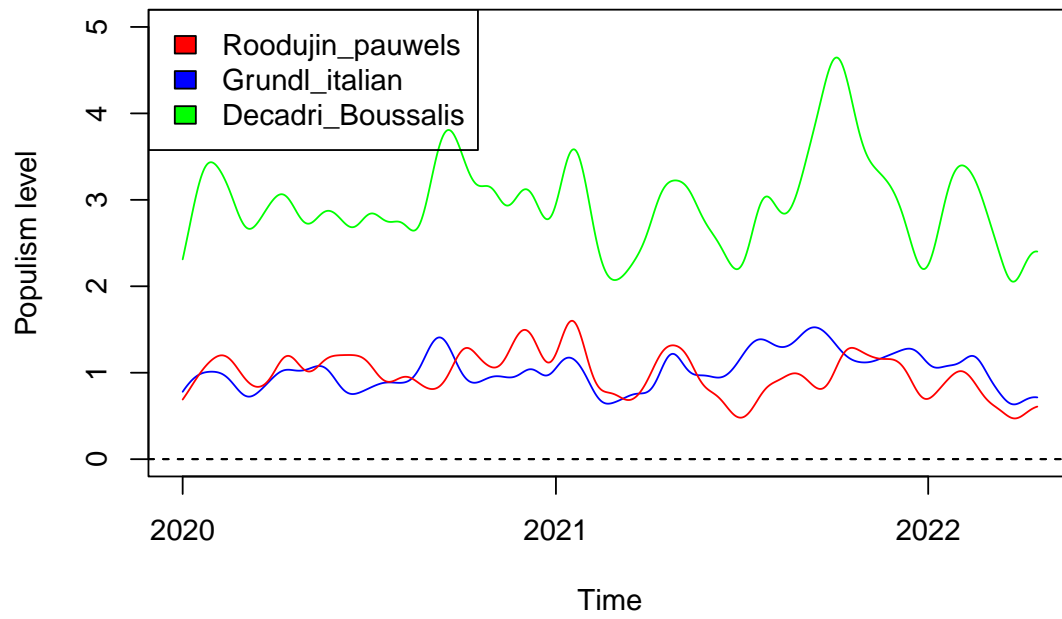


3.7 Compare the general level of populism in time for the dictionaries

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

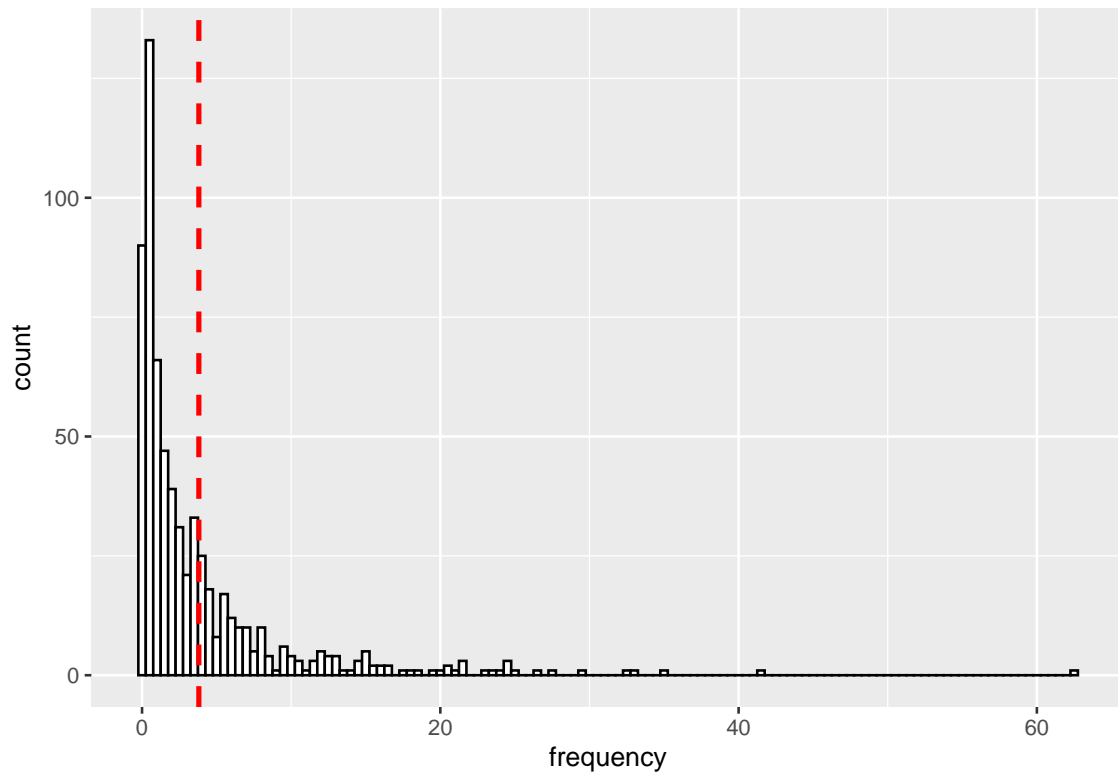
```
## integer(0)
```

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

```
# TEST
```

```
ggplot(dict_4_tstat_nome, aes(x=frequency)) +  
  geom_histogram(binwidth=.5, colour="black", fill="white") +  
  geom_vline(aes(xintercept=mean(frequency, na.rm=T)), # Ignore NA values for m  
             color="red", linetype="dashed", size=1)
```



4 Sentiment analysis

<http://saifmohammad.com/WebPages/lexicons.html>

4.1 Inspect the dictionary

```
head(get_sentiment_dictionary(dictionary = "nrc", language = "italian"), 15)
```

##	lang	word	sentiment	value
## 1	italian	abba	positive	1
## 2	italian	capacità	positive	1
## 3	italian	sopra citato	positive	1
## 4	italian	assoluto	positive	1
## 5	italian	assoluzione	positive	1
## 6	italian	assorbito	positive	1
## 7	italian	abbondanza	positive	1
## 8	italian	abbondante	positive	1
## 9	italian	accademico	positive	1
## 10	italian	accademia	positive	1
## 11	italian	accettabile	positive	1
## 12	italian	accettazione	positive	1
## 13	italian	accessibile	positive	1
## 14	italian	encomio	positive	1
## 15	italian	alloggio	positive	1

4.1.1 Clean text from dataframe

Define function to make the text extracted from dataframe suitable for analysis

```

# Define function to make the text suitable for analysis
clean.text = function(x)
{
  # tolower
  x = tolower(x)
  # remove rt
  x = gsub("rt", "", x)
  # remove at
  x = gsub("@\\w+", "", x)
  # remove punctuation
  x = gsub("[:punct:]", "", x)
  # remove numbers
  x = gsub("[:digit:]", "", x)
  # remove links http
  x = gsub("http\\w+", "", x)
  # remove tabs
  x = gsub("[ \\t]{2,}", "", x)
  # remove blank spaces at the beginning
  x = gsub("^ ", "", x)
  # remove blank spaces at the end
  x = gsub(" $", "", x)
  return(x)
}

```

4.2 Create the filtered dataframes

```
# Create filtered dataframes
```

```
MELONI <- dataset %>% filter(nome == "MELONI Giorgia")  
CONTE <- dataset %>% filter(nome == "CONTE Giuseppe")  
RENZI <- dataset %>% filter(nome == "RENZI Matteo")  
SALVINI <- dataset %>% filter(nome == "SALVINI Matteo")  
LETTA <- dataset %>% filter(nome == "LETTA Enrico")  
BERLUSCONI <- dataset %>% filter(nome == "BERLUSCONI Silvio")  
SPERANZA <- dataset %>% filter(nome == "SPERANZA Roberto")
```

4.3 Create nrc objects

```
# Create the nrc object
```

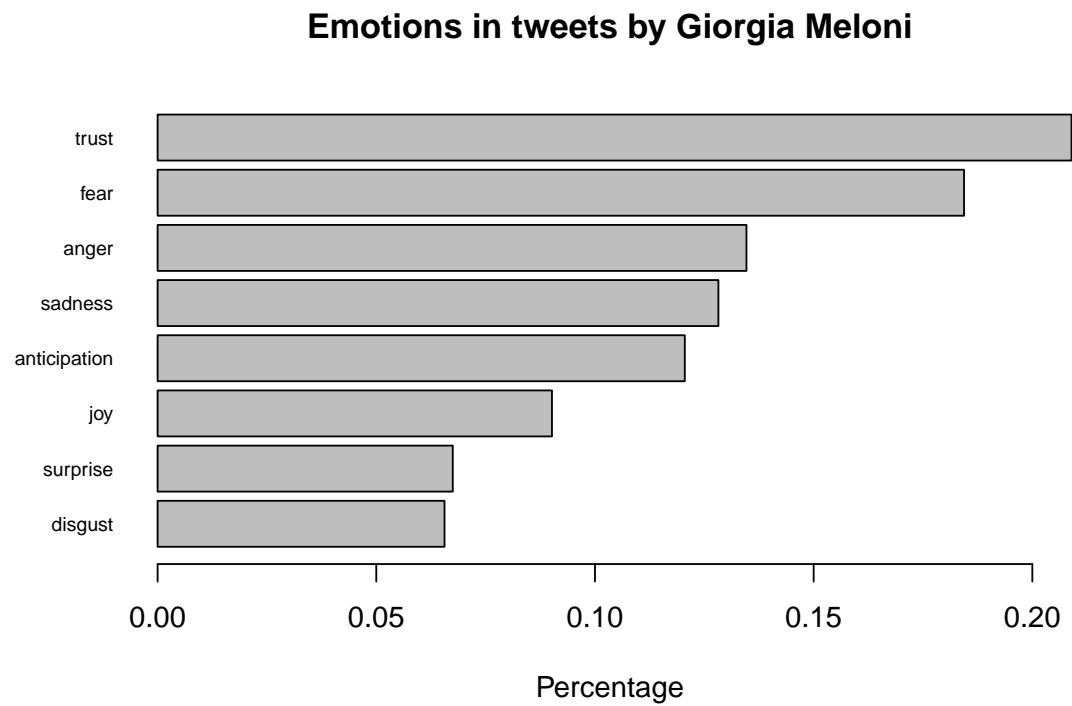
```
nrc_meloni <- get_nrc_sentiment(MELONI$tweet_testo, language="italian")  
save(nrc_meloni, file="data/nrc_meloni.Rda")  
  
nrc_conte <- get_nrc_sentiment(CONTE$tweet_testo, language="italian")  
save(nrc_conte, file="data/nrc_conte.Rda")  
  
nrc_renzi <- get_nrc_sentiment(RENZI$tweet_testo, language="italian")  
save(nrc_renzi, file="data/nrc_renzi.Rda")  
  
nrc_salvini <- get_nrc_sentiment(SALVINI$tweet_testo, language="italian")  
save(nrc_salvini, file="data/nrc_salvini.Rda")  
  
nrc_letta <- get_nrc_sentiment(LETTA$tweet_testo, language="italian")  
save(nrc_letta, file="data/nrc_letta.Rda")  
  
nrc_berlusconi <- get_nrc_sentiment(BERLUSCONI$tweet_testo, language="italian")
```

```
save(nrc_berlusconi, file="data/nrc_berlusconi.Rda")

nrc_speranza <- get_nrc_sentiment(SPERANZA$tweet_testo, language="italian")
save(nrc_speranza, file="data/nrc_speranza.Rda")
```

4.4 Giorgia Meloni

4.4.1 Proportion of the emotion



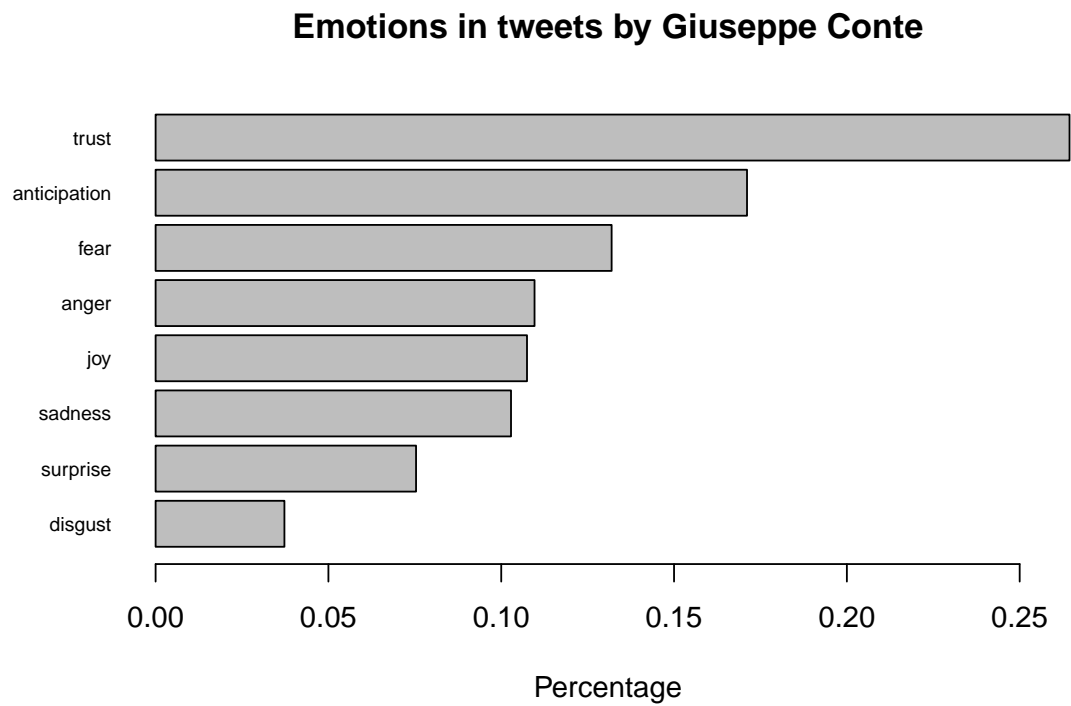
4.4.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Giorgia Meloni



4.5 Giuseppe Conte

4.5.1 Proportion of the emotion



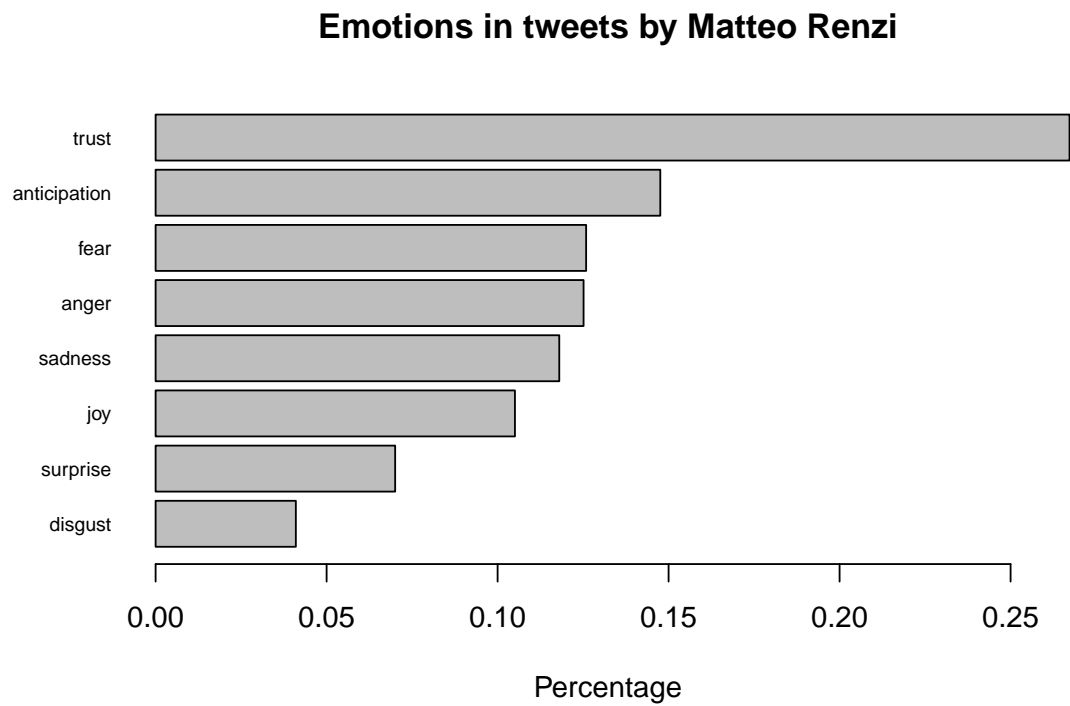
Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Giuseppe Conte



4.6 Matteo Renzi

4.6.1 Proportion of the emotion



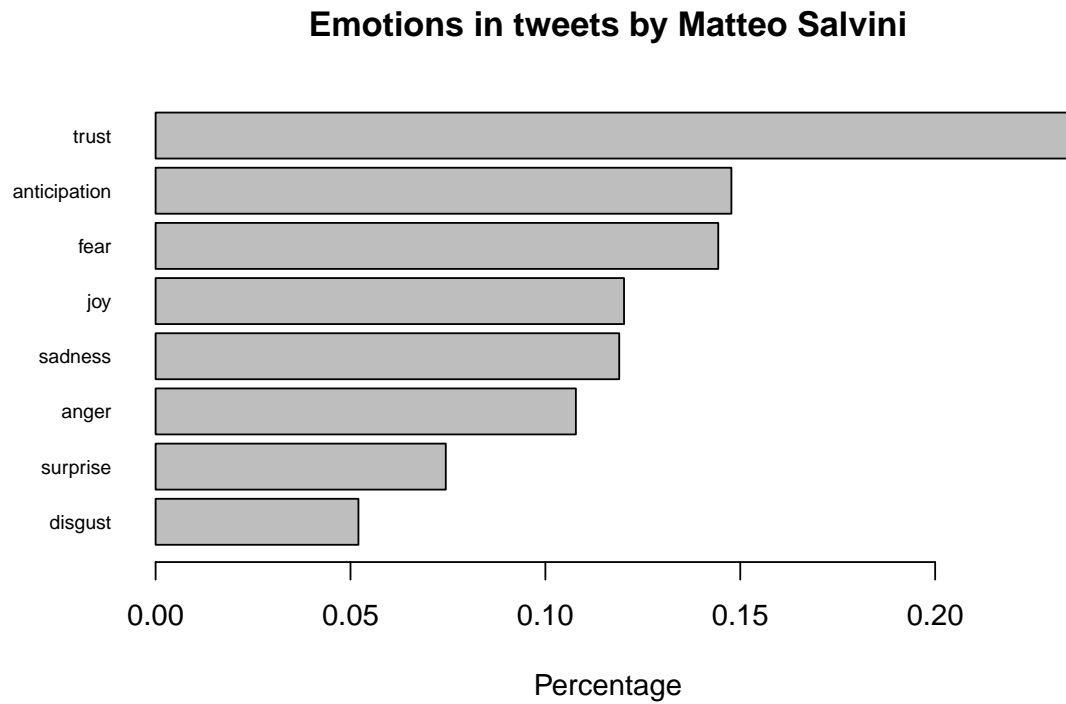
4.6.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Matteo Renzi



4.7 Matteo Salvini

4.7.1 Proportion of the emotion



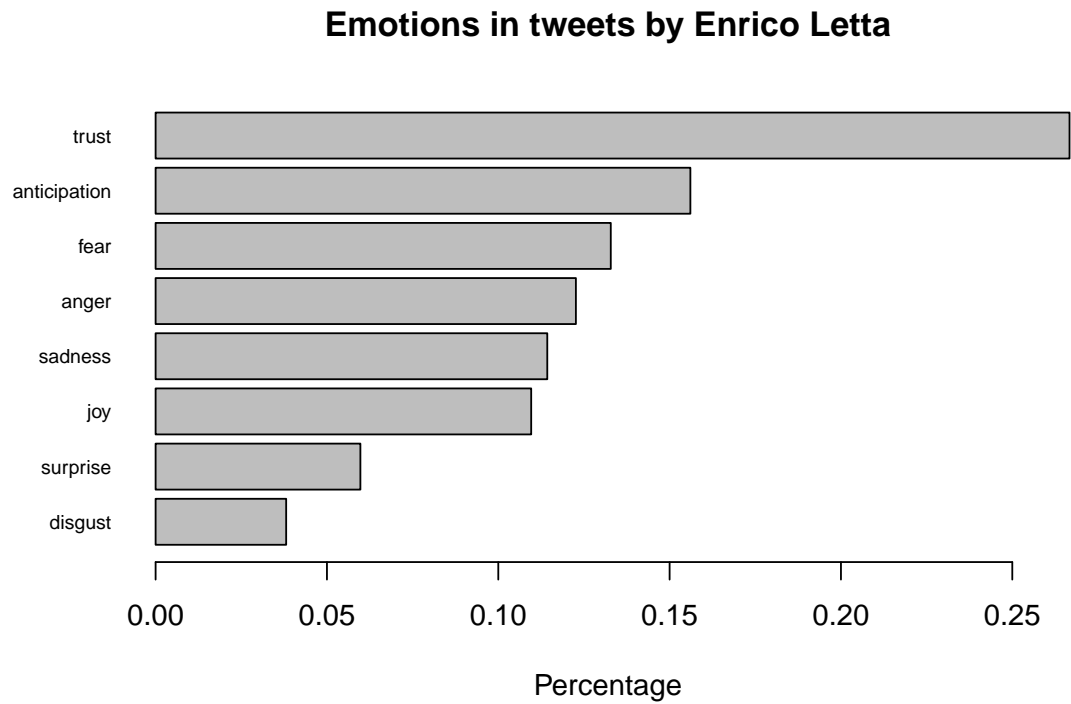
4.7.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Matteo Salvini



4.8 Enrico Letta

4.8.1 Proportion of the emotion



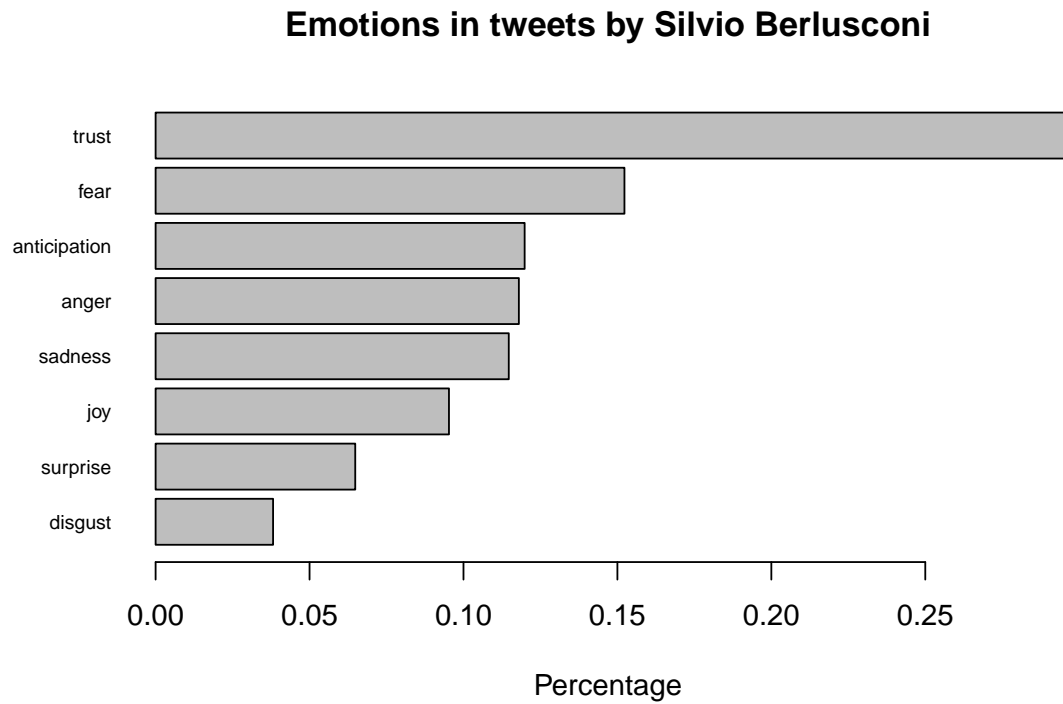
4.8.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Enrico Letta



4.9 Silvio Berlusconi

4.9.1 Proportion of the emotion



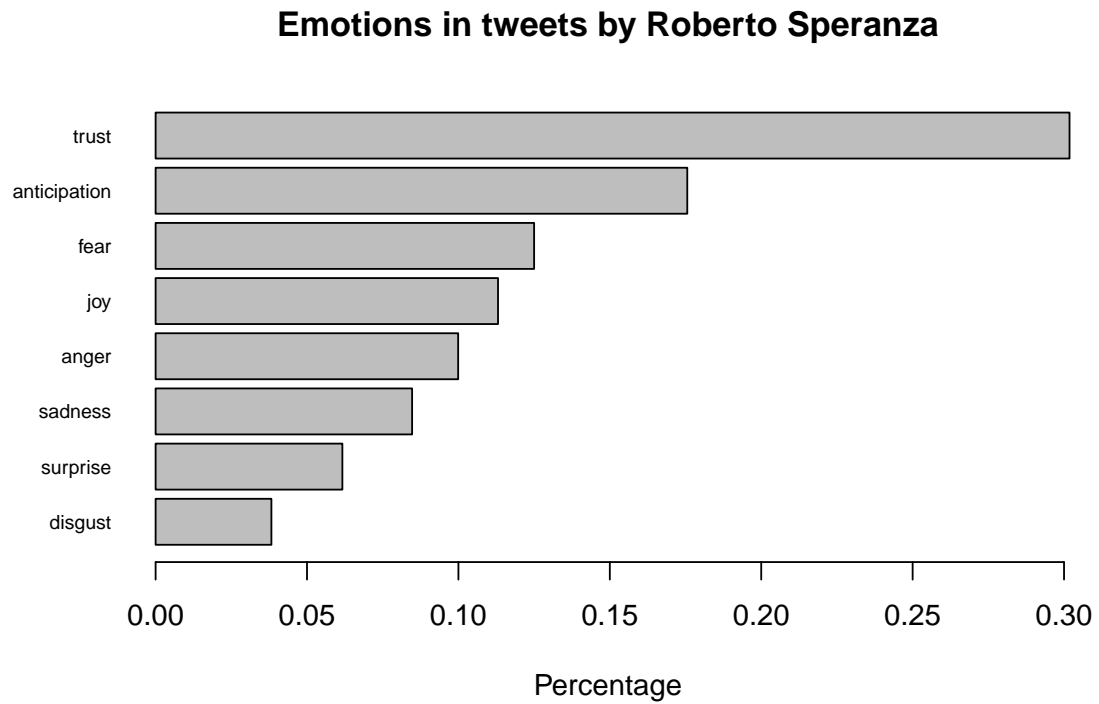
4.9.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Silvio Berlusconi



4.10 Roberto Speranza

4.10.1 Proportion of the emotion



4.10.2 Wordcloud of emotions

Emotion Comparison Word Cloud for tweets by Roberto Speranza



5 LDA Topic model analysis

5.1 CREATE THE DTM

5.1.1 Remove all the account's mentions

```
DFM_trimmed@Dimnames$features <- gsub("^@", "", DFM_trimmed@Dimnames$features)
```

Convert the Document Feature Matrix (Dfm) in a Topic Model (Dtm)

```
dtm <- quanteda::convert(DFM_trimmed, to = "topicmodels")
```

5.2 FIND THE BEST NUMBER OF TOPICS K

5.2.1 Search the best number of Topics comparing coherence and exclusivity values

K = 10:50

```
# 10 : 50 iter 1000
```

```
top1 <- c(10:50)
```

```
## let's create an empty data frame
```

```
risultati <- data.frame(first=vector(), second=vector(), third=vector())
```

```
system.time(
```

```
  for (i in top1)
```

```
  {
```

```
    set.seed(123)
```

```
    lda_test <- LDA(dtm, method= "Gibbs", k = (i), control=list(verbose=50L, iter=
```

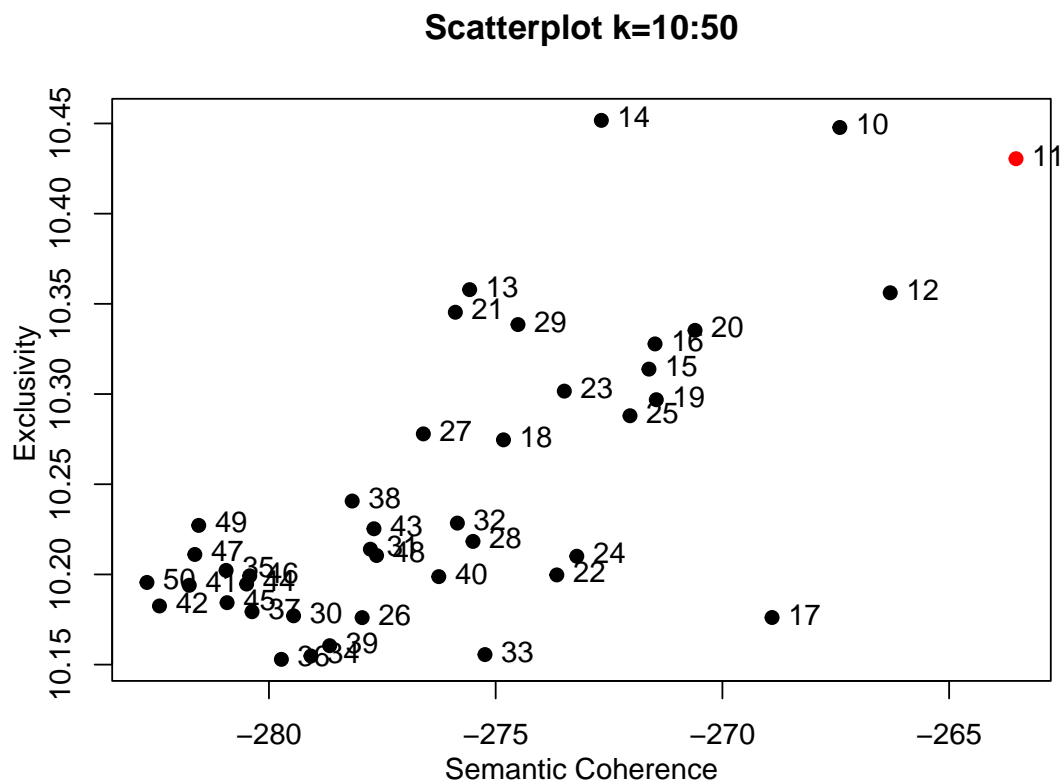
```

topic <- (i)
coherence_test <- mean(topic_coherence(lda_test, dtm))
exclusivity_test <- mean(topic_exclusivity(lda_test))
risultati <- rbind(risultati , cbind(topic, coherence_test, exclusivity_test ))
}
)

# save(risultati,file="data/results_K_10-50.Rda")

```

5.2.2 Plot the values of coherence and exclusivity in order to find the best K



Interpreting those data i can state that K= 11 has the best values of coherence and exclusivity.

5.3 ANALISYS OF THE TOPICS

5.3.1 Repeat the analysis selecting $k = 11$

```
system.time(lda_11 <- LDA(dtm, method= "Gibbs", k = 11, control = list(seed = 123))  
# save(lda_11, file = "data/lda_k_11.Rda")
```

5.3.2 The most important terms from the model, for each topic

Top terms 01	Top terms 02	Top terms 03	Top terms 04	Top terms 05	Top terms 06
governo	#coronavirus	diretta	forza_italia	solidarietà	grande
italiani	#covid19	roma	bene	libertà	grazie
lega	sicurezza	domani	cose	parole	mondo
conte	covid	città	vero	diritti	italia
matteosalvinimi	scuola	amici	davvero	rispetto	forza
fratelliditalia	salute	sindaco	ragione	democrazia	de
salvini	pandemia	insieme	problema	diritto	italiano
pd	pass	sera	ce	violenza	italiana
#lega	dati	territorio	parla	guerra	storia
casa	green	parlare	dovrebbe	popolo	l'italia

Top terms 7	Top terms 8	Top terms 9	Top terms 10	Top terms 11
lavoro	via	governo	anni	piano
paese	ministro	imprese	grazie	nazionale
presidente	legge	milioni	donne	futuro
politica	parlamento	euro	giornata	importante
buon	commissione	lavoratori	auguri	sociale
l'italia	senato	famiglie	vittime	giovani
momento	mov5stelle	sostegno	famiglia	europea
bene	voto	cittadini	comunità	fondamentale
pdnetwork	camera	misure	servizio	paesi
insieme	intervento	crisi	forze	intervista

```
titles_11 <- c("1", "2", "3", "4", "5", "6",  
              "7", "8", "9", "10", "11")
```

```
table_titles_11 <- rbind (titles_11, terms)
```

```
t11.1 <- table_titles_11[,1:6]
```

```
t11.2 <- table_titles_11[,7:11]
```

```
kable(t11.1)%>%
```

```
  kable_styling(latex_options = c("HOLD_position","scale_down"))
```

	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 6
titles_11	1	2	3	4	5	6
	governo	#coronavirus	diretta	forza_italia	solidarietà	grande
	italiani	#covid19	roma	bene	libertà	grazie
	lega	sicurezza	domani	cose	parole	mondo
	conte	covid	città	vero	diritti	italia
	matteosalvinimi	scuola	amici	davvero	rispetto	forza
	fratelliditalia	salute	sindaco	ragione	democrazia	de
	salvini	pandemia	insieme	problema	diritto	italiano
	pd	pass	sera	ce	violenza	italiana
	#lega	dati	territorio	parla	guerra	storia
	casa	green	parlare	dovrebbe	popolo	l'italia

```
kable(t11.2)%>%
```

```
  kable_styling(latex_options = c("HOLD_position","scale_down"))
```


	Topic 7	Topic 8	Topic 9	Topic 10	Topic 11
titles_11	7	8	9	10	11
	lavoro	via	governo	anni	piano
	paese	ministro	imprese	grazie	nazionale
	presidente	legge	milioni	donne	futuro
	politica	parlamento	euro	giornata	importante
	buon	commissione	lavoratori	auguri	sociale
	l'italia	senato	famiglie	vittime	giovani
	momento	mov5stelle	sostegno	famiglia	europea
	bene	voto	cittadini	comunità	fondamentale
	pdnetwork	camera	misure	servizio	paesi
	insieme	intervento	crisi	forze	intervista

6 FER: Facial Emotion Recognition Analysis

6.1 Report on the analysis made with FER Puthon package

The package use the FER-2013 dataset created by Pierre Luc Carrier and Aaron Courville.

The dataset was created using the Google image search API to search for images of faces that match a set of 184 emotion-related keywords like “blissful”, “enraged,” etc. These keywords were combined with words related to gender, age or ethnicity, to obtain nearly 600 strings which were used as facial image search queries. The first 1000 images returned for each query were kept for the next stage of processing. OpenCV face recognition was used to obtain bounding boxes around each face in the collected images. Human labelers than rejected incorrectly labeled images, corrected the cropping if necessary, and filtered out some duplicate images. Approved, cropped images were then resized to 48x48 pixels and converted to grayscale. Mehdi Mirza and Ian Goodfellow prepared a subset of the images for this contest, and mapped the fine-grained emotion keywords into the same seven broad categories used in the Toronto Face Database [Joshua Susskind, Adam Anderson, and Geoffrey E. Hinton. The Toronto face dataset. Technical Report UTML TR 2010-001, U. Toronto, 2010.]. The resulting dataset contains 35887 images, with 4953 “Anger” images, 547 “Disgust” images, 5121 “Fear” images, 8989 “Happiness” images, 6077 “Sadness” images, 4002 “Surprise” images, and 6198 “Neutral” images. FER-2013 could theoretical suffer from label errors due to the way it was collected, but Ian Goodfellow found that human accuracy on FER-2013 was $65\pm 5\%$.

66% ACCURACY REPORTED BY OCTAVIO ARRIAGA, Matias Valdenegro-Toro, Paul Plöger (Real-time Convolutional Neural Networks for Emotion and Gender Classification)