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 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

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

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

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

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

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

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

1.4 Create the month variable

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

1.4.1 Check the number of month

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

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

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

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

1.5 Create the trimester variable

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

1.5.1 Check the number of trimesters

```
max(tw$quarter)
```

```
## [1] 10
```

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

```
## [1] 10
```

1.6 Create the year variables

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

1.6.1 Check the number of years

```
max(tw$year)
```

```
## [1] 3
```

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

```
## [1] 3
```

1.7 Count the number of missing values

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

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

1.7.1 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$month)),
sum(is.na(tw$quarter)),
sum(is.na(tw$year)))

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
quarter	0
year	0

From this check I'll obtain 148178 urls missing, this variable is not collected properly and we will not use in the analysis, and also results 6494 tweets missings, those are the cases when someone post only images or video without text, so the extraction is correct.

1.7.2 Remove rows with missing tweets

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

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

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

1.8 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"
```

1.8.1 Adjust the variable genere

```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)
```

1.8.2 Verify 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

1.9 Create a new dataset selecting only necessary informations

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

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

1.10 Create the corpus

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

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

1.11 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)
```

1.12 Remove the emoji

```
# Create a copy of the dfm
test <- DFM
# 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$features)
b <- unique(DFM@Dimnames$features)
setdiff(b,a) #I have selected also words that must not be removed

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

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

#save(DFM,file="data/dfm.Rda")
```

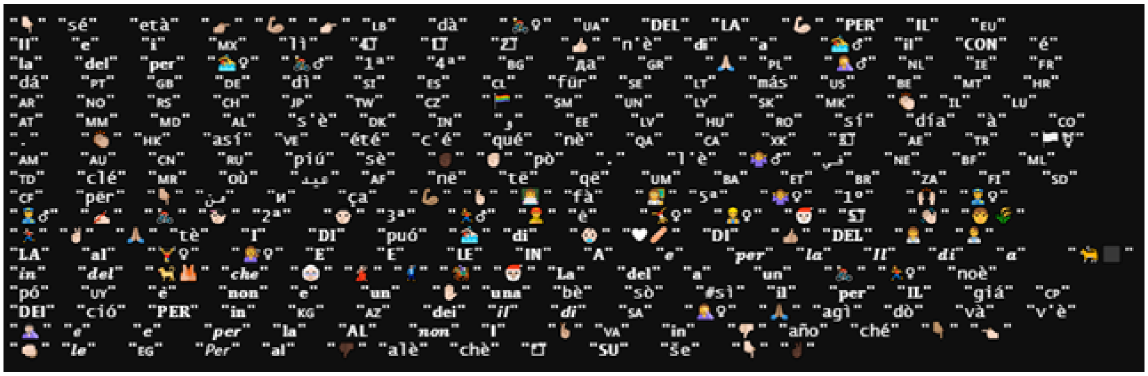


Figure 1: Emoji removed

1.12.1 Now the data are ready for the next analysis

2 Preliminar analysis

2.1 Who is inside this dataset?

```
# Number of parliamentarians
n_parl <- length(unique(dataset$nome))
n_parl
```

```
## [1] 730
```

```
# How many parliamentarians for each party_id?
n_parl_party <- dataset %>% select(party_id, nome) %>% group_by(party_id) %>% unique
kable(n_parl_party)
```

party_id	n
CI	17
FDI	39
FI	96
INDIPENDENTE	6
IV	5
LEGA	134
LEU	15
M5S	197
MISTO	71
PD	144
REG_LEAGUES	7

```
# Gender composition
n_gender <- dataset %>% select(genere, nome) %>% group_by(genere) %>% unique() %>% c
kable(n_gender)
```

genere	n
female	258
male	472

```
# Wich is the period of analysis?
max(tw$date)
```

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

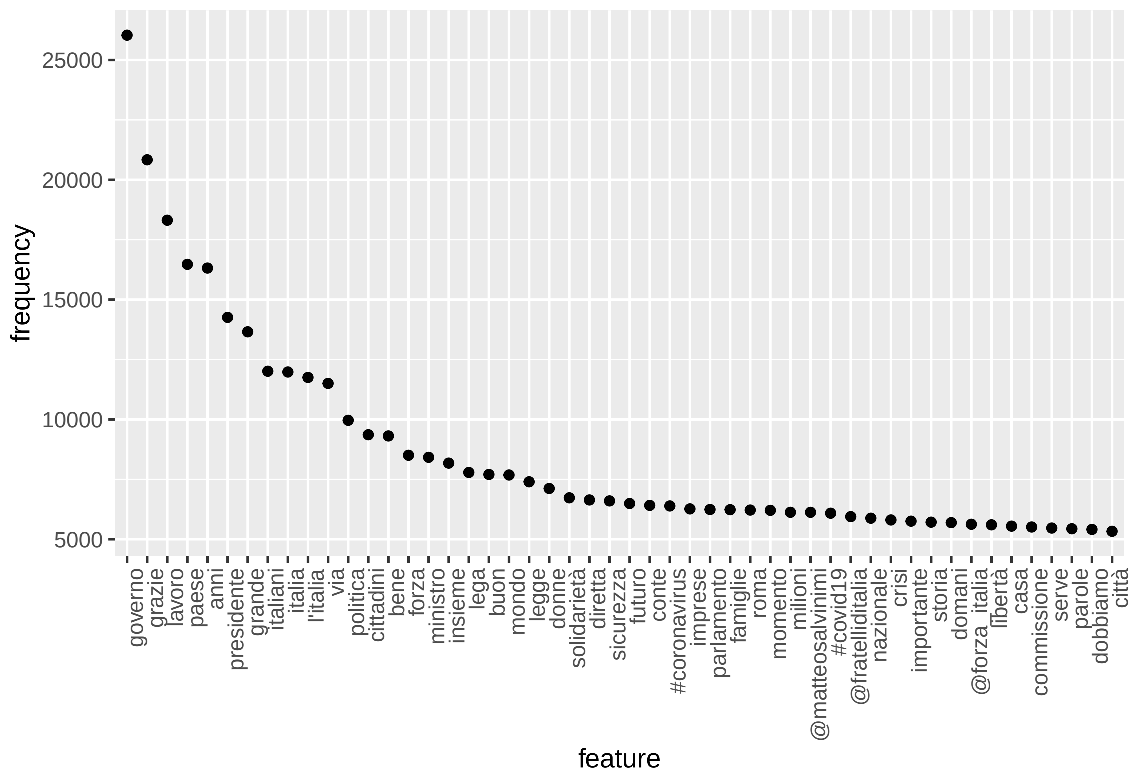
```
min(tw$date)
```

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

2.2 Topfeatures frequency

```
# Textplotwordcloud
set.seed(123)
textplot_wordcloud(DFM, min_count = 20,max_words = 300,
     color = c('red', 'pink', 'green', 'purple', 'blue'))
```

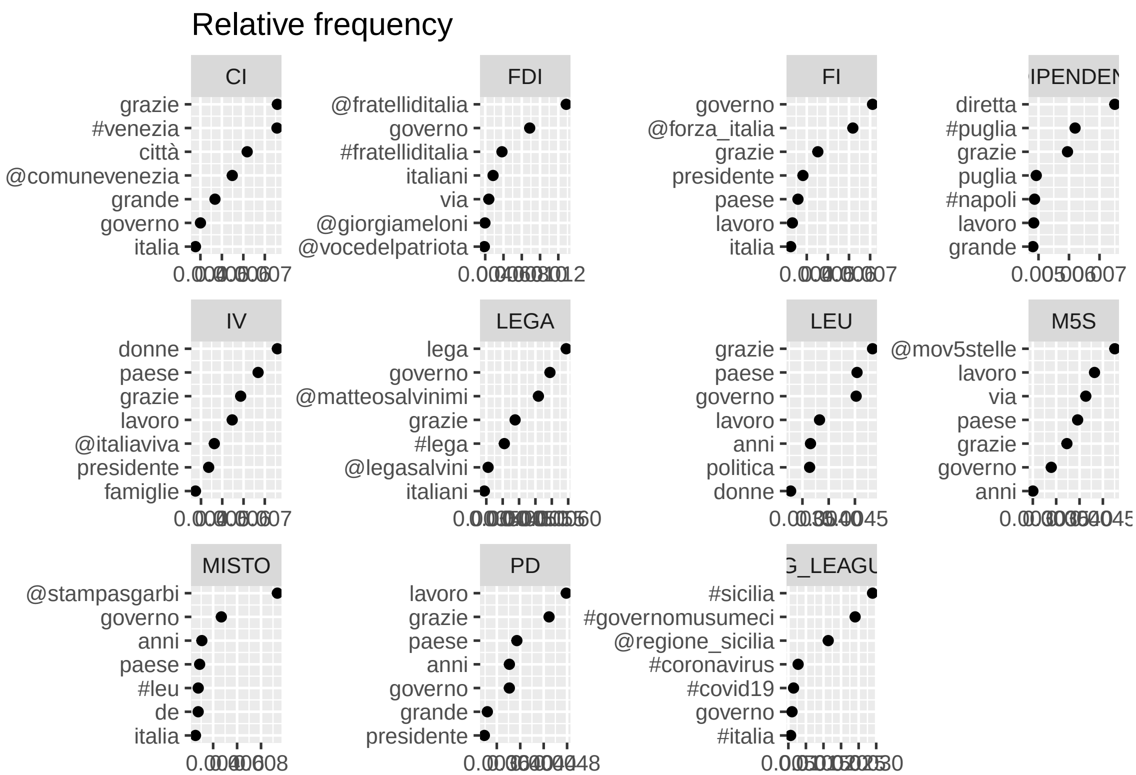

```r
# Plot frequency of the topfeatures in the DFM
features_dfm <- textstat_frequency(DFM, n = 50)
# Sort by reverse frequency order
features_dfm$feature <- with(features_dfm, reorder(feature, -frequency))
ggplot(features_dfm, aes(x = feature, y = frequency)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



2.2.1 Relative frequency of the topfeatures by Party ID

```r
# group and weight the DFM
dfm_party_weight <- dfm_group(DFM, groups = party_id) %>%
  dfm_weight(scheme = "prop")

# Plot relative frequency by party_id
freq_weight <- textstat_frequency(dfm_party_weight, n = 7, groups = 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) +
  ggtitle("Relative frequency") +
  labs(x = NULL, y = NULL)
```

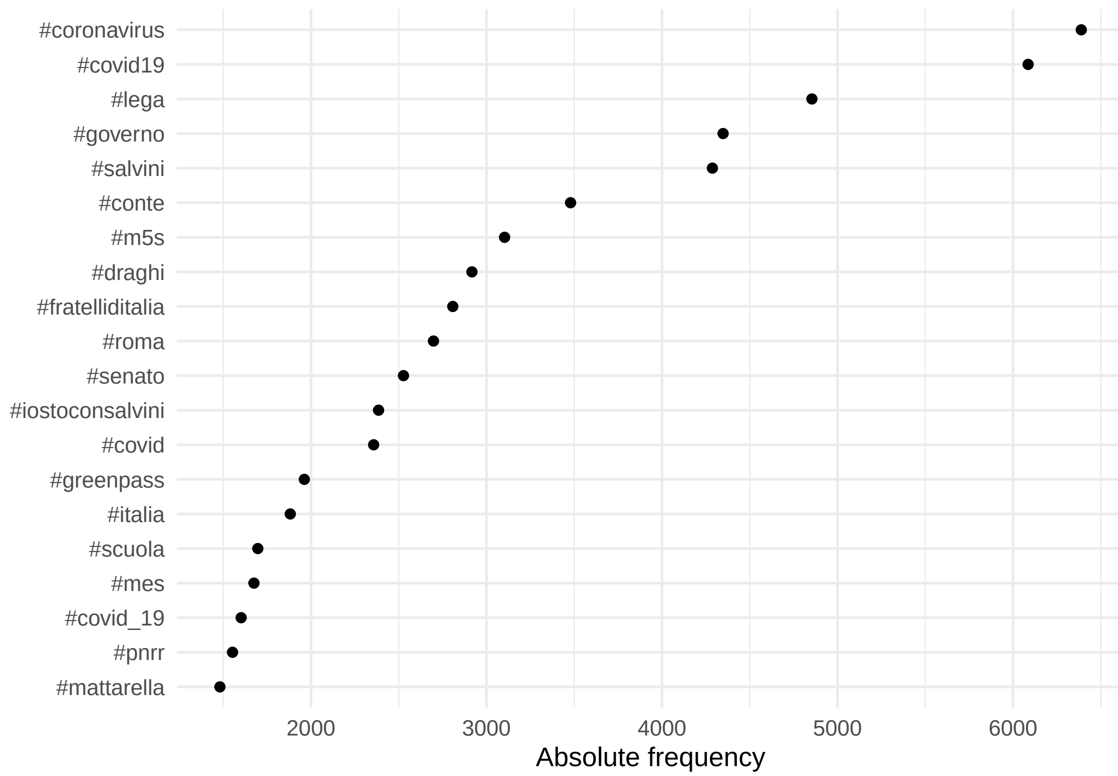


2.3 Most common hashtag

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

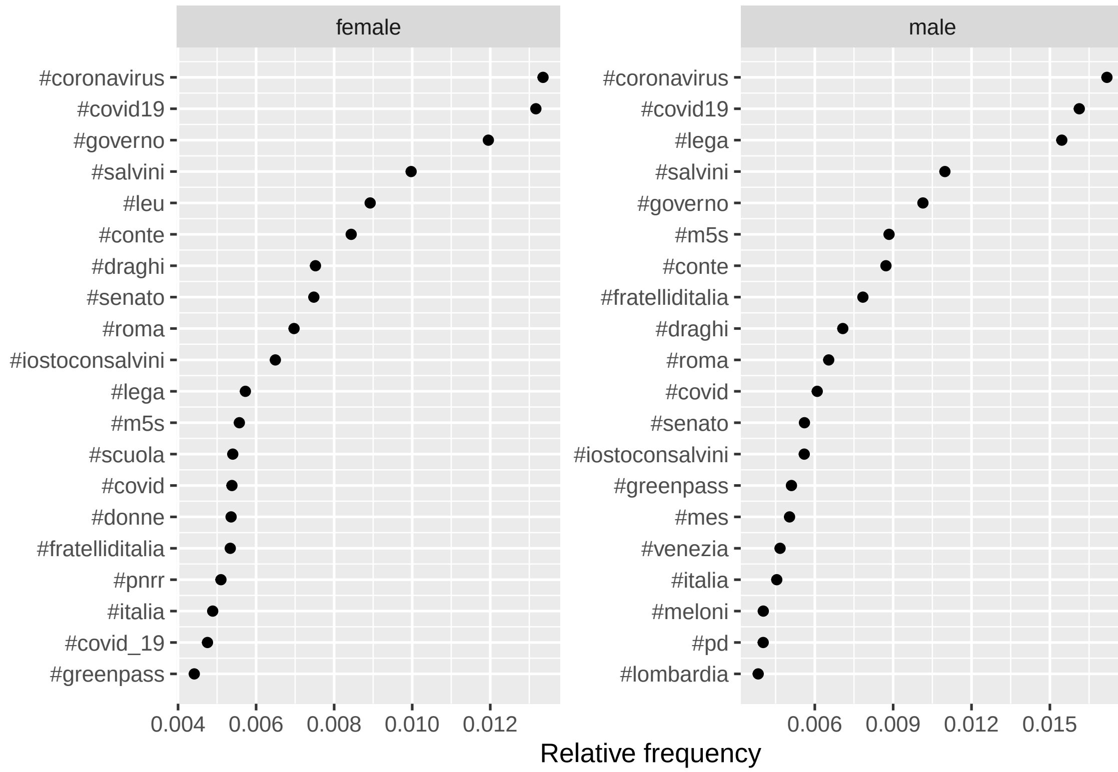
```
##  [1] "#coronavirus"     "#covid19"           "#lega"              "#governo"
```

```
##  [5] "#salvini"          "#conte"            "#m5s"              "#draghi"
##  [9] "#fratelliditalia" "#roma"             "#senato"           "#iostoconsalvini"
## [13] "#covid"           "#greenpass"        "#italia"           "#scuola"
## [17] "#mes"             "#covid_19"         "#pnrr"             "#mattarella"
```



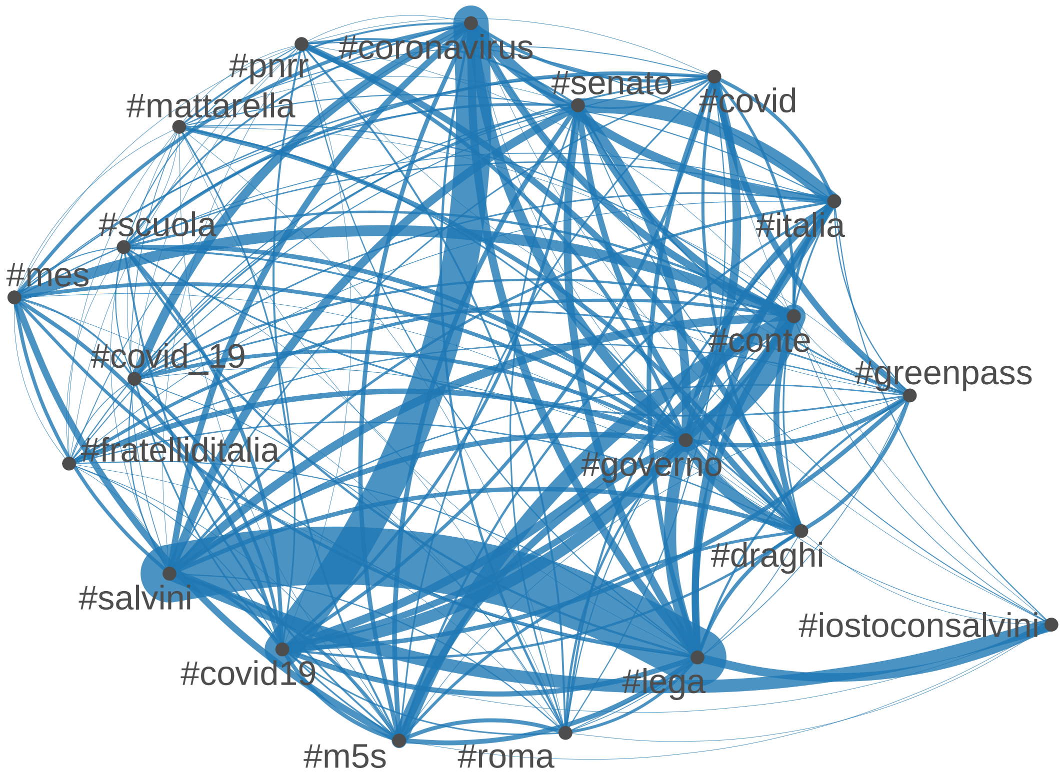
2.3.1 Most common hashtag by Gender

```r
# group and weight the DFM
dfm_gender_weight <- dfm_group(tag_dfm, groups = genere) %>%
  dfm_weight(scheme = "prop")

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



2.3.2 Co-occurrence Plot of hashtags

```
# 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)
```



2.4 Most frequently mentioned usernames

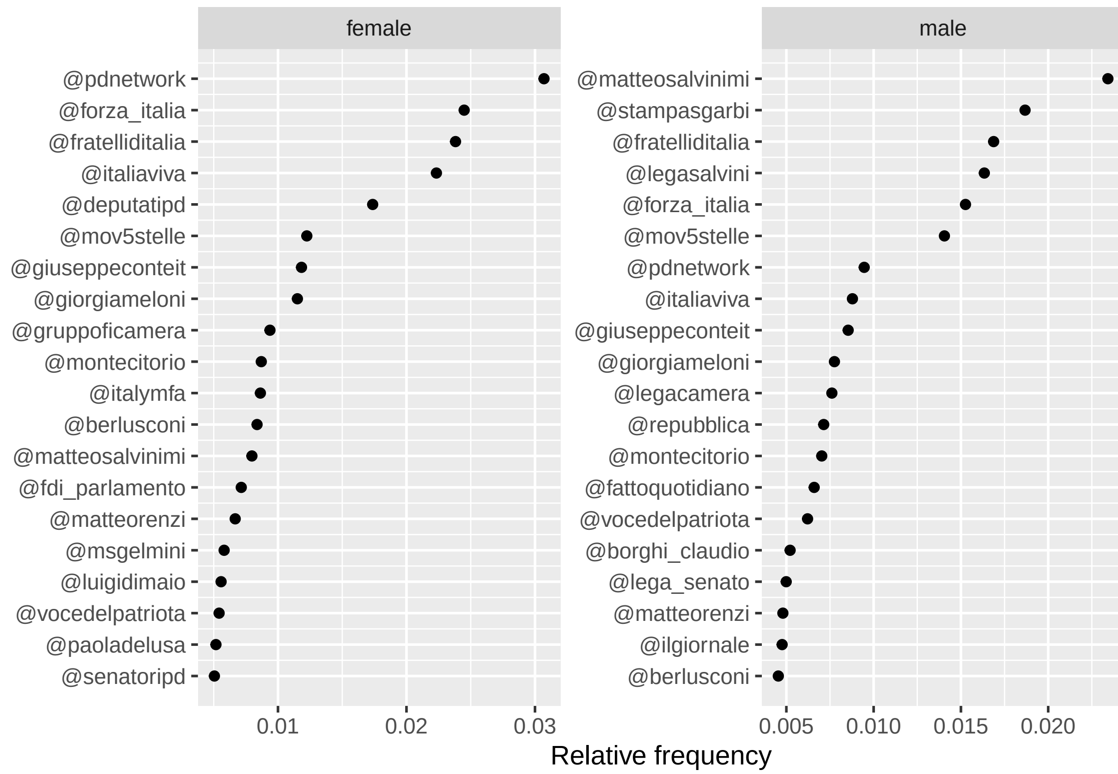
```
user_dfm <- dfm_select(DFM, pattern = "@*")
topuser <- names(topfeatures(user_dfm, 20, scheme = "docfreq"))
kable(topuser, col.names = "Most mentioned username")
```

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

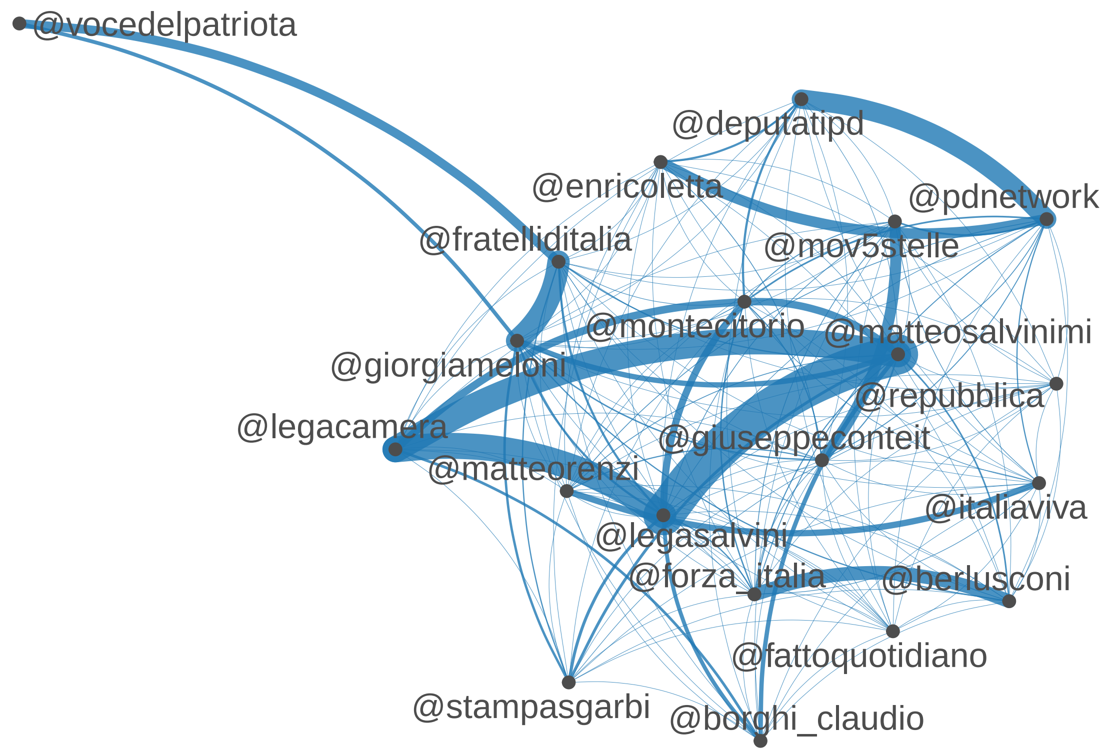
2.4.1 Most frequently mentioned usernames by gender

```
# group and weight the DFM
user_dfm_gender_weight <- dfm_group(user_dfm, groups = genere) %>%
  dfm_weight(scheme = "prop")
```

```r
user_tstat_freq <- textstat_frequency(user_dfm_gender_weight, n = 20,
                                      groups = user_dfm_gender_weight$genere)
```



2.4.2 Co-occurrence plot of usernames



2.5 How many times a politician cite his/her party

```
party_citations <- data.frame(first = vector(), second = vector() )
system.time(
for (i in unique(tw$party_id))
{
  a <- paste("#", i ,sep = "")
  b <- tw %>% filter(grepl(a,tweet_testo)&party_id== i) %>% count()
  c <- tw %>% filter(party_id == i) %>% count()
  d <- (b/c) * 100
  party_citations <- rbind(party_citations, cbind(i,b,c,d))
```

```
}
)

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

```
colnames(party_citations) <- c("party","n_citations", "Number of tweets", "perc")

kable(party_citations %>% arrange(desc(perc)),
      col.names = c("Party","Number of citations",
                    "number of tweets", "% of citations"))
```

Party	Number of citations	number of tweets	% of citations
M5S	1581	54418	2.9052887
LEGA	511	87162	0.5862647
FDI	131	36177	0.3621085
PD	179	91997	0.1945716
IV	5	3129	0.1597955
FI	62	65264	0.0949988
CI	1	6954	0.0143802
REG_LEAGUES	0	1398	0.0000000
MISTO	0	34644	0.0000000
INDIPENDENTE	0	2186	0.0000000
LEU	0	7868	0.0000000

In the above script i search the # for the parliamentary group, but is very unlikely, for example, that someone use the #IV for talking about the "Italia Viva" party, so i decided to enrich the dataframe creating a new variable with the name of the official twitter page for every party, and repeat the search using it.

I created the variable party_Page for only those parliamentary group that has a direct connection with a party (i excluded Reg_leagues, misto and indipendente)

2.5.1 Create the variable with the name of the official Twitter account

```
tw <- tw %>% mutate(party_page = if_else(party_id == "PD", "@pdnetwork",
if_else( party_id == "FDI", "@FratellidItalia",
    if_else(party_id == "M5S", "@Mov5Stelle",
      if_else(party_id == "FI", "@forza_italia",
        if_else(party_id == "LEGA", "@LegaSalvini",
            if_else(party_id == "CI", "@coraggio_italia",
              if_else(party_id == "LEU", "@liberi_uguali",
                                                "NA"))))))))
```

2.5.2 Count for each party how many times a politician cite their respective party

```
party_citations_page <- data.frame(first = vector(), second = vector(),
                                  third = vector(), fourth = vector())
system.time(
  for (i in unique(tw$party_page))
  {
    b <- tw %>% filter(grepl(i,tweet_testo)&party_page== i) %>% count()
    c <- tw %>% filter(party_page == i) %>% count()
    d <- (b/c) * 100
    party_citations_page <- rbind(party_citations_page, cbind(i,b,c,d))

  }
)
# save(party_citations_page, file = "data/party_citations_page.Rda")
```

```
colnames(party_citations_page) <- c("party","n_citations",
                                    "Number of tweets", "perc")

kable(party_citations_page %>% filter(party != "NA") %>%
        arrange(desc(perc)),
      col.names = c("Party","Number of citations",
                    "number of tweets", "% of citations"))
```

Party	Number of citations	number of tweets	% of citations
@FratellidItalia	5842	36177	16.1483816
@forza_italia	5203	65264	7.9722358
@Mov5Stelle	3873	54418	7.1171304
@ItaliaViva	201	3129	6.4237776
@pdnetwork	4194	91997	4.5588443
@LegaSalvini	3364	87162	3.8594800
@coraggio_italia	131	6954	1.8838079
@liberi_uguali	16	7868	0.2033554

2.6 How many times the party leader is cited by his/her party

2.6.1 Create the variable with the official leader's account for every party

```
tw <- tw %>% mutate(party_leader =
if_else(party_id == "PD" & date < "2021-03-14", "@nzingaretti",
if_else(party_id == "PD" & date > "2021-03-14", "@EnricoLetta",
if_else( party_id == "FDI", "@GiorgiaMeloni",
if_else(party_id == "M5S" &date < "2020-01-22" , "@luigidimaio",
```

```
if_else(party_id == "M5S" &date > "2020-01-22" &date < "2021-08-06", "@vitocrimi",
if_else(party_id == "M5S" & date > "2021-08-061", "@GiuseppeConteIT",
if_else(party_id == "FI", "@berlusconi",
if_else(party_id == "LEGA", "@matteosalvinimi",                              i
if_else(party_id == "CI", "@LuigiBrugnaro",
if_else(party_id == "LEU", "@robersperanza",
"NA")))))))))))
```

2.6.2 Count for each party how many times a politician cite his/ her party leader

```
leader_citations <- data.frame(first = vector(), second = vector(),
                               third = vector(), fourth = vector())
system.time(
  for (i in unique(tw$party_leader))
  {
    b <- tw %>% filter(grepl(i,tweet_testo)&party_leader== i) %>% count()
    c <- tw %>% filter(party_leader == i) %>% count()
    d <- (b/c) * 100
    leader_citations <- rbind(leader_citations, cbind(i,b,c,d))


  }
)
 #save(leader_citations, file = "data/leader_citations.Rda")
```

```
colnames(leader_citations) <- c("leader","n_citations",
                                "Number of tweets", "perc")
```

```r
kable(leader_citations %>% filter(leader != "NA") %>%
        arrange(desc(perc)),
      col.names = c("Leader","Number of citations",
                    "Number of tweets", "% of citations"))
```

Leader	Number of citations	Number of tweets	% of citations
@matteosalvinimi	4826	87162	5.5368165
@GiorgiaMeloni	1745	36177	4.8235066
@GiuseppeConteIT	444	15517	2.8613778
@luigidimaio	30	1184	2.5337838
@berlusconi	1533	65264	2.3489213
@EnricoLetta	709	44520	1.5925427
@matteorenzi	46	3129	1.4701182
@nzingaretti	475	47305	1.0041222
@robersperanza	45	7868	0.5719370
@vitocrimi	107	37544	0.2849989
@LuigiBrugnaro	19	6954	0.2732240

2.7 How many times a politician cite itself in the tweet

```r
self_citations <- data.frame(first = vector(), second = vector() )
system.time(
for (i in unique(tw$tw_screen_name))
{
  a <- paste("@", i ,sep = "")
  b <- tw %>% filter(grepl(a,tweet_testo) & tw_screen_name== i) %>% count()
  c <- tw %>% filter(tw_screen_name == i) %>% count()
  d <- (b/c) * 100
  self_citations <- rbind(self_citations, cbind(i,b,c,d))
```

```r
}
)
#save(self_citations, file = "data/self_citations.Rda")
```

```r
colnames(self_citations) <- c("Politician","n_citations",
                              "Number of tweets", "perc")


kable(self_citations %>% filter(n_citations > 2) %>%
        arrange(desc(perc)),
      col.names = c("Politician","Number of citations",
                    "Number of tweets", "% of citations"))
```

Politician	Number of citations	Number of tweets	% of citations
wandaferro1	32	55	58.1818182
FrassinettiP	32	163	19.6319018
albertlaniece	51	282	18.0851064
Luca_Sut	20	341	5.8651026
DalilaNesci	17	341	4.9853372
PatassiniTullio	13	714	1.8207283
matteodallosso	3	170	1.7647059
sbonaccini	33	2884	1.1442441
sfnlcd	9	1308	0.6880734
gianluc_ferrara	3	560	0.5357143
adolfo_urso	7	1966	0.3560529
gualtierieurope	4	1432	0.2793296
MassimoUngaro	3	1135	0.2643172
EugenioGiani	3	1235	0.2429150
pierofassino	3	1255	0.2390438
ecdelre	4	2113	0.1893043
guglielmopicchi	3	3234	0.0927644

3 Dictionary analysis

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

I use 3 dictionaries 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.
- Grundl: Gründl J. Populist ideas on social media: A dictionary-based measurement of populist communication. New Media & Society. December 2020.
- 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.
- This previous dictionary is used in the version called "Decadri & Boussalis + Grundl": that is simply a more extended version of the D&B dictionary, which also contains some terms taken from Grundl.

3.1 Create the dictionary

I imported the excel file with the words for the dictionaries, excluding NA's.

```
# 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_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)]))
```

```r
dictionaries <- c("Rooduijn_Pauwels_Italian", "Grundl_Italian_adapted"
                  ,"Decadri_Boussalis_Grundl")
n.words <- c(
  length(Rooduijn_Pauwels_Italian$populism),
  length(Grundl_Italian_adapted$populism),
  (length(Decadri_Boussalis_Grundl$people)+
     length(Decadri_Boussalis_Grundl$common_will)+
     length(Decadri_Boussalis_Grundl$elite))
            )

number_of_words <- data.frame(dictionaries,n.words)

kable(number_of_words)
```

dictionaries	n.words
Rooduijn_Pauwels_Italian	18
Grundl_Italian_adapted	135
Decadri_Boussalis_Grundl	77

3.1.1 Group and weight the dfm

```r
# By party & quarter
dfm_weigh_p_quart <- dfm_group(DFM, groups = interaction(party_id, quarter))%>%
  dfm_weight(scheme = "prop")
```

Apply the dictionaries

3.2 Decadri_Boussalis_Grundl

```
# Dictionary analysis with Decadri_Boussalis_Grundl
# By quarter
dfm_dict1  <- dfm_lookup(dfm_weigh_p_quart, dictionary = Decadri_Boussalis_Grundl)
```

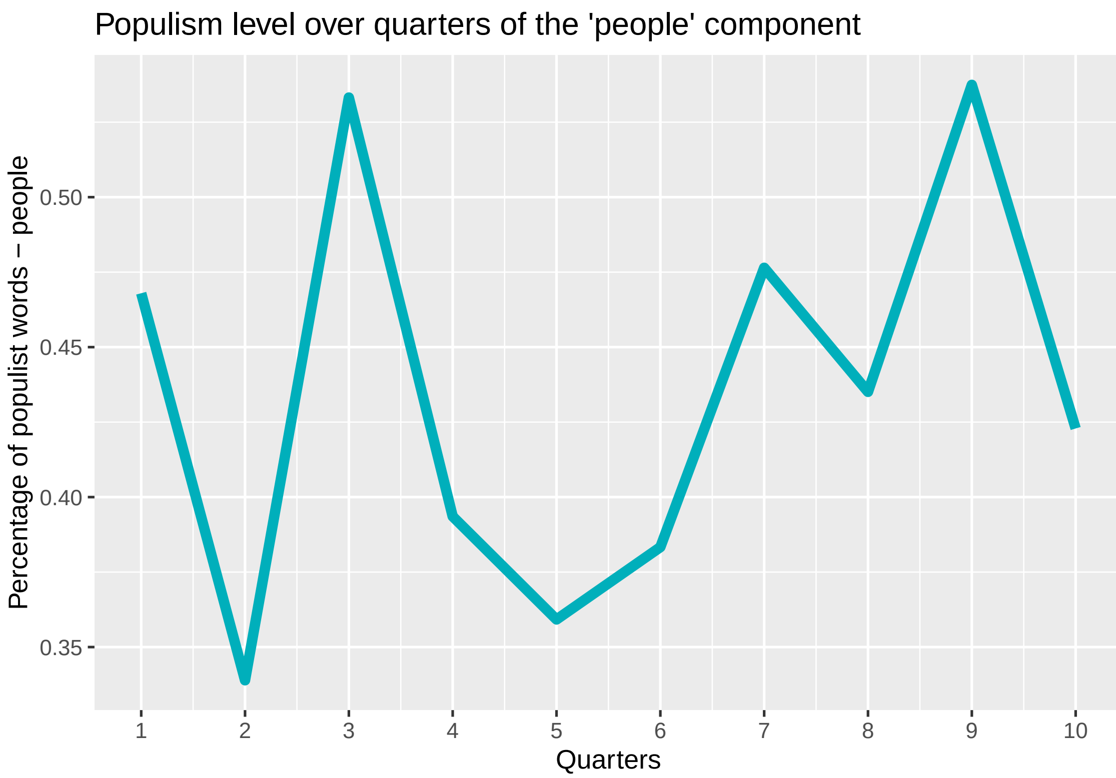
3.2.1 Transform the DFM into an ordinary dataframe

```
data_dict1 <- dfm_dict1 %>%
  quanteda::convert(to = "data.frame") %>%
  cbind(docvars(dfm_dict1))

# Add variable with general level of populism
data_dict1 <- data_dict1 %>% mutate(populism = (people + common_will + elite) * 100)
```

3.2.2 Level of populism in time

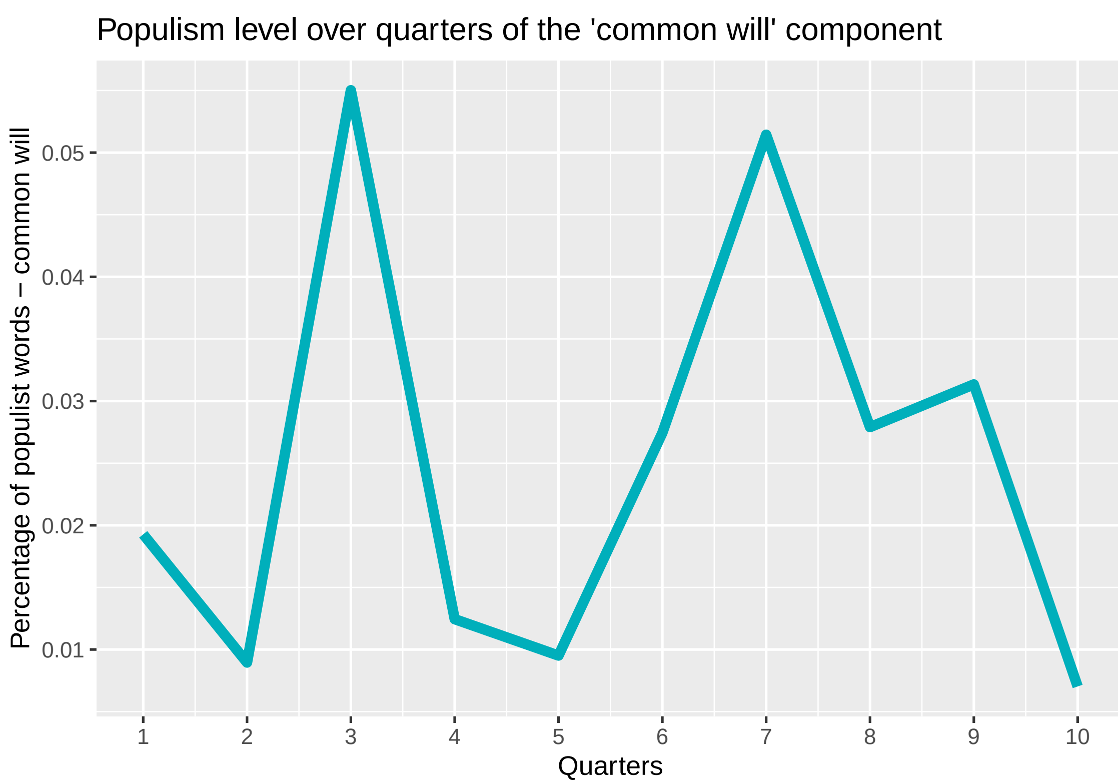
```
#Over time PEOPLE (quarters)
data_quarter_people <- aggregate(x = data_dict1$people,  # Specify data column
          by = list(data_dict1$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_people$perc <- data_quarter_people$x * 100

# plot the level of the "people" component in time
plot_people <- ggplot(data = data_quarter_people, aes(x = Group.1, y = perc))+
```

```
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_people$Group.1)
  ylab("Percentage of populist words - people")+
  labs(title = "Populism level over quarters of the 'people' component")
plot_people
```




```
###########
#Over time COMMON WILL (quarters)
data_quarter_common <- aggregate(x = data_dict1$common_will,  # Specify data column
          by = list(data_dict1$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_common$perc <- data_quarter_common$x * 100


# plot the level of the "common will" component in time
```

```r
plot_common <- ggplot(data = data_quarter_common, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_common$Group.1)
  ylab("Percentage of populist words - common will")+
  labs(title = "Populism level over quarters of the 'common will' component")
plot_common
```

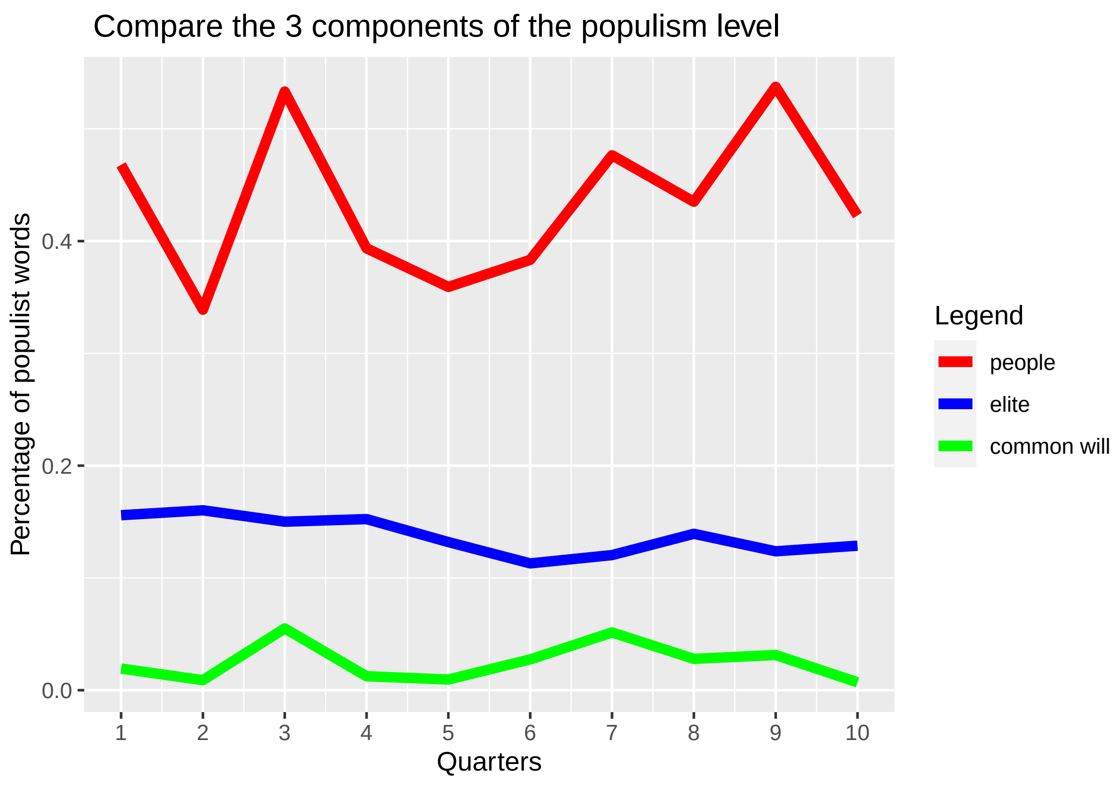



```r
###########
#Over time ELITE (quarters)
data_quarter_elite <- aggregate(x = data_dict1$elite,  # Specify data column
          by = list(data_dict1$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_elite$perc <- data_quarter_elite$x * 100
```

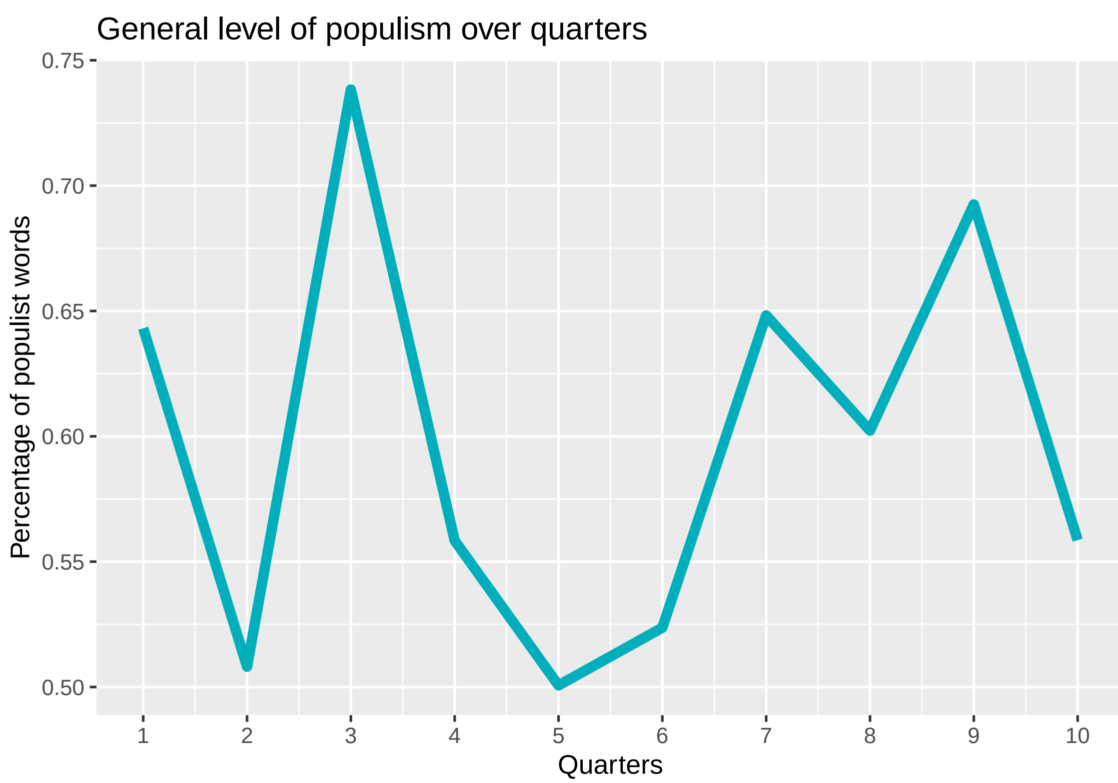
```r
# plot the level of the "ELITE" component in time
plot_elite <- ggplot(data = data_quarter_elite, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_elite$Group.1),
  ylab("Percentage of populist words - elite")+
  labs(title = "Populism level over quarters of the 'elite' component")
plot_elite
```


```r
#########
# compare the levels
p <- ggplot() +
  # plot people
  geom_line(data = data_quarter_people, aes(x = Group.1, y = perc, color = "people")
  # plot common will
```

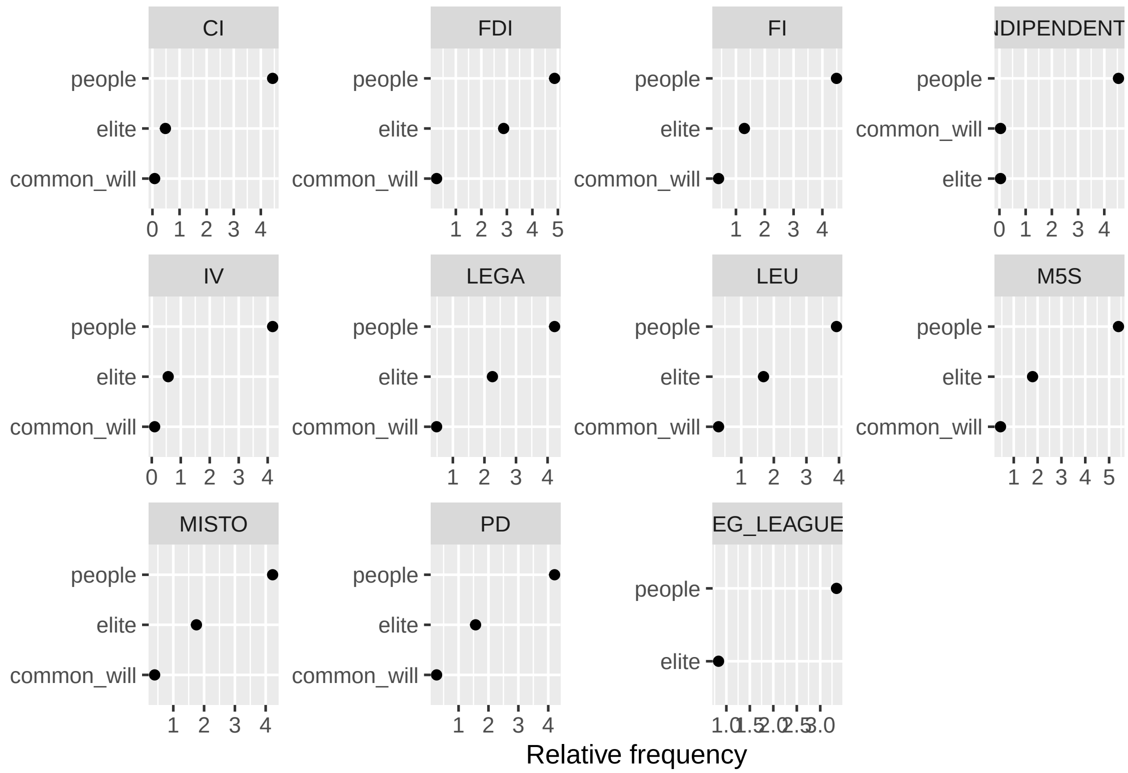
```
  geom_line(data = data_quarter_common, aes(x = Group.1, y = perc, color = "common w
  # plot elite
  geom_line(data = data_quarter_elite, aes(x = Group.1, y = perc, color = "elite"),
  scale_color_manual(name='Legend',
                     breaks=c('people', 'elite', 'common will'),
                     values=c('people'='red', 'elite'='blue', 'common will'='green')

  scale_x_continuous("Quarters", labels = as.character(data_quarter_people$Group.1)
  ylab("Percentage of populist words")+
  labs(title = " Compare the 3 components of the populism level")

p
```



```r
################
#Over time general level populism (quarters)
data_quarter_general <- aggregate(x = data_dict1$populism,  # Specify data column
          by = list(data_dict1$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_general$perc <- data_quarter_general$x

# plot the level of populism
plot_general <- ggplot(data = data_quarter_general, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_general$Group.1)
  ylab("Percentage of populist words")+
  labs(title = "General level of populism over quarters")
plot_general
```



3.2.3 Frequencies of the 3 components of populism for each parliamentary group

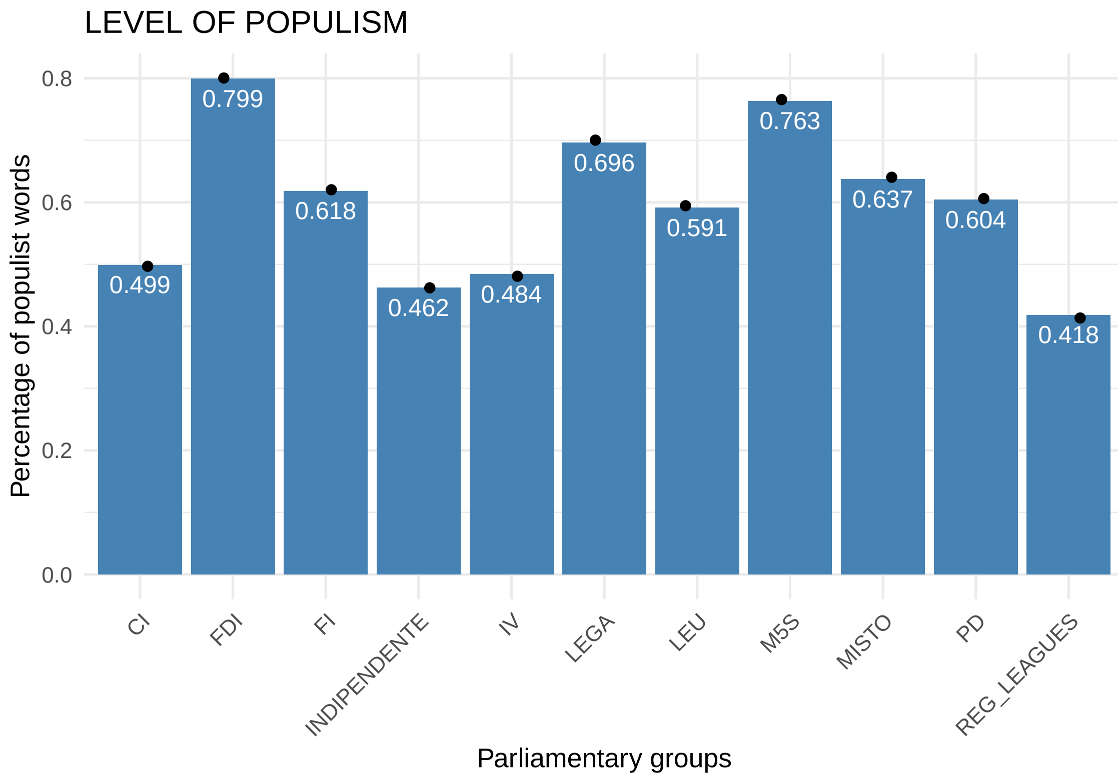


3.2.4 Ranking of parliamentary groups according to their level of populism

```
#By party no time (quarters)

# POPULISM
data_party <- aggregate(x = data_dict1$populism,  # Specify data column
          by = list(data_dict1$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party$perc <- round(data_party$x,3)
kable(data_party %>% select(Group.1, perc) %>%  arrange(desc(perc)))
```

Group.1	perc
FDI	0.799
M5S	0.763
LEGA	0.696
MISTO	0.637
FI	0.618
PD	0.604
LEU	0.591
CI	0.499
IV	0.484
INDIPENDENTE	0.462
REG_LEAGUES	0.418

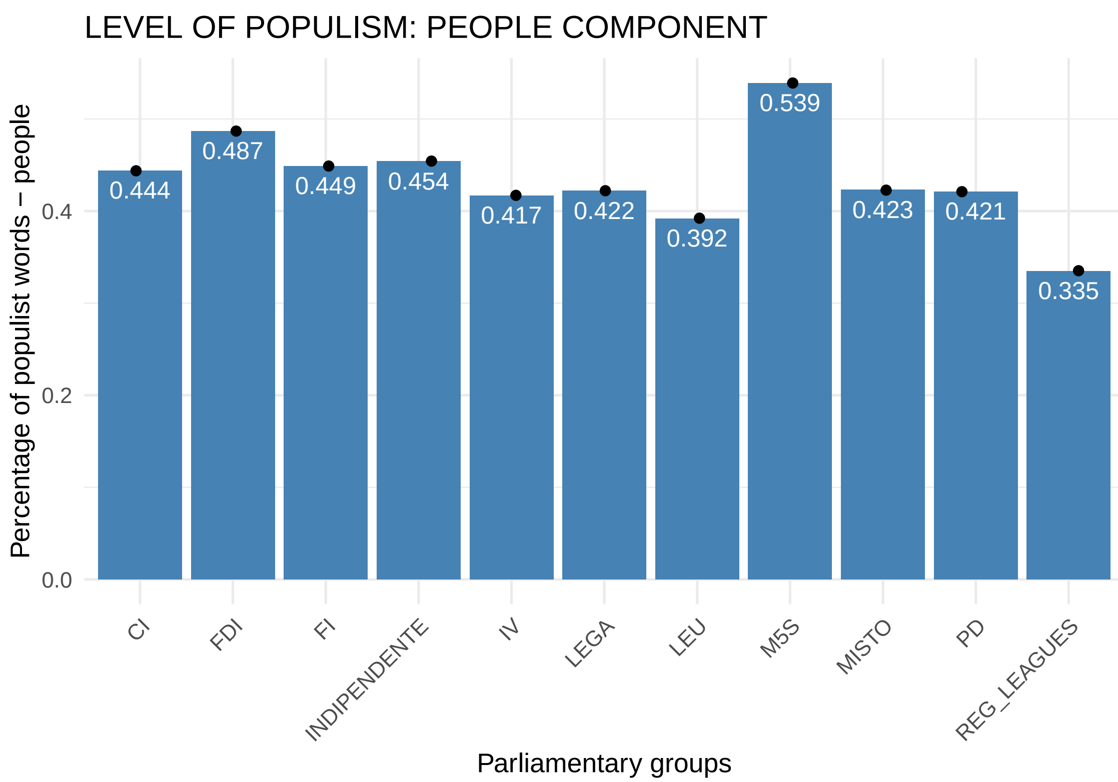
```
ggplot(data=data_party, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  geom_jitter(width=0.15)+
  theme(axis.text.x = element_text(angle = 45, hjust=1))+
  ylab("Percentage of populist words") +
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM")
```




```
# PEOPLE
data_party_people <- aggregate(x = data_dict1$people,  # Specify data column
          by = list(data_dict1$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party_people$perc <- round(data_party_people$x * 100,3)
kable(data_party_people %>% select(Group.1, perc)%>% arrange(desc(perc)))
```

Group.1	perc
M5S	0.539
FDI	0.487
INDIPENDENTE	0.454
FI	0.449
CI	0.444
MISTO	0.423
LEGA	0.422
PD	0.421
IV	0.417
LEU	0.392
REG_LEAGUES	0.335

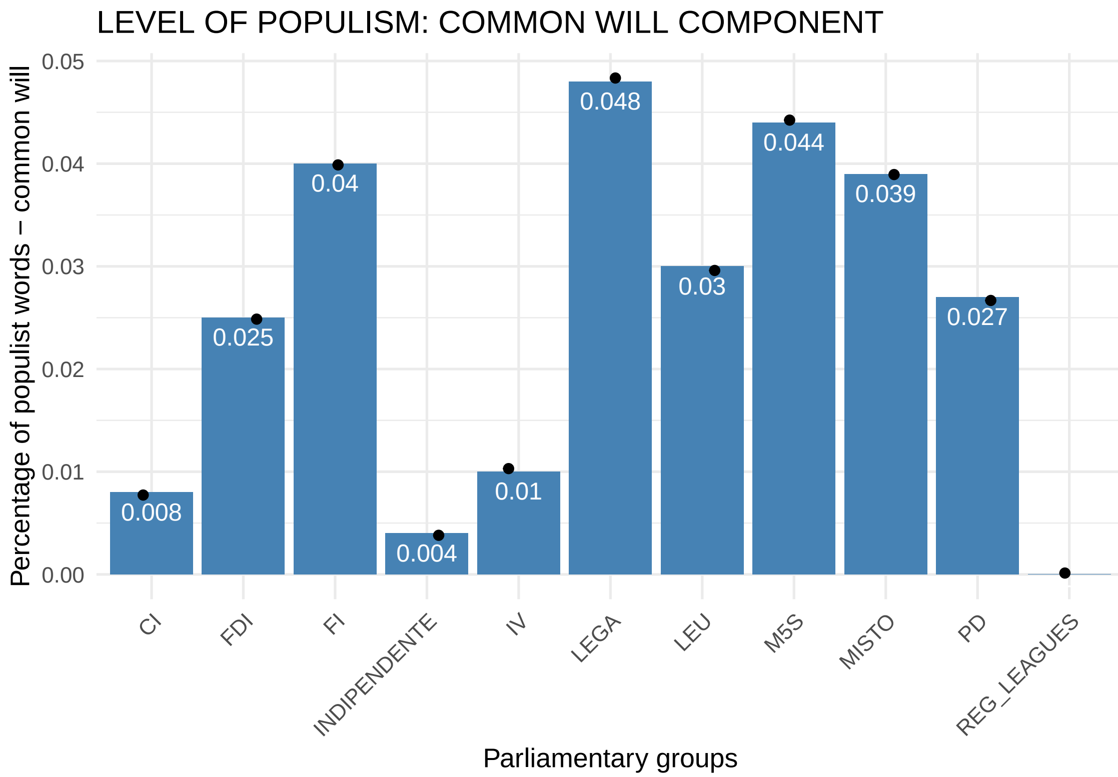
```
ggplot(data=data_party_people, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  geom_jitter(width=0.15)+
  theme(axis.text.x = element_text(angle = 45, hjust=1))+
  ylab("Percentage of populist words - people")+
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM: PEOPLE COMPONENT")
```




```
# COMMON WILL
data_party_common <- aggregate(x = data_dict1$common_will,  # Specify data column
          by = list(data_dict1$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party_common$perc <- round(data_party_common$x * 100,3)
kable(data_party_common %>% select(Group.1, perc)%>% arrange(desc(perc)))
```

Group.1	perc
LEGA	0.048
M5S	0.044
FI	0.040
MISTO	0.039
LEU	0.030
PD	0.027
FDI	0.025
IV	0.010
CI	0.008
INDIPENDENTE	0.004
REG_LEAGUES	0.000

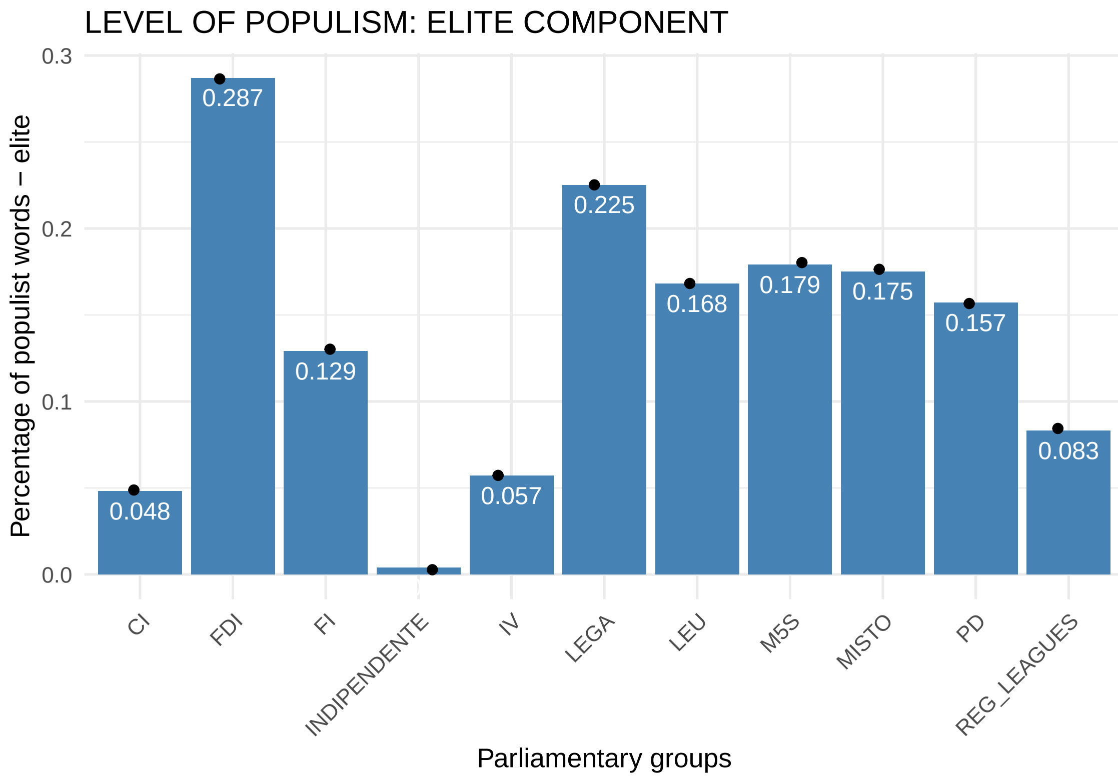
```
ggplot(data=data_party_common, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  geom_jitter(width=0.15)+
  theme(axis.text.x = element_text(angle = 45, hjust=1))+
  ylab("Percentage of populist words - common will")+
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM: COMMON WILL COMPONENT")
```




```r
# ELITE
data_party_elite <- aggregate(x = data_dict1$elite,  # Specify data column
          by = list(data_dict1$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party_elite$perc <- round(data_party_elite$x * 100,3)
kable(data_party_elite %>% select(Group.1, perc)%>% arrange(desc(perc)))
```

Group.1	perc
FDI	0.287
LEGA	0.225
M5S	0.179
MISTO	0.175
LEU	0.168
PD	0.157
FI	0.129
REG_LEAGUES	0.083
IV	0.057
CI	0.048
INDIPENDENTE	0.004

```
ggplot(data=data_party_elite, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  geom_jitter(width=0.15)+
  theme(axis.text.x = element_text(angle = 45, hjust=1))+
  ylab("Percentage of populist words - elite")+
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM: ELITE COMPONENT")
```



Are the average values of populism for each party statistically different from each other? The reference category is PD

```
# bivariate regression for check t-test
data_dict1$factor_party <- as.factor(data_dict1$party_id)
data_dict1$factor_party <- relevel(data_dict1$factor_party, ref = "PD")

data_dict1$factor_quarter <- as.factor(data_dict1$quarter)
data_dict1$factor_quarter <- relevel(data_dict1$factor_quarter, ref = "8")

a3 <- lm(populism ~  factor_quarter + factor_party, data_dict1 )

summary(a3)
```

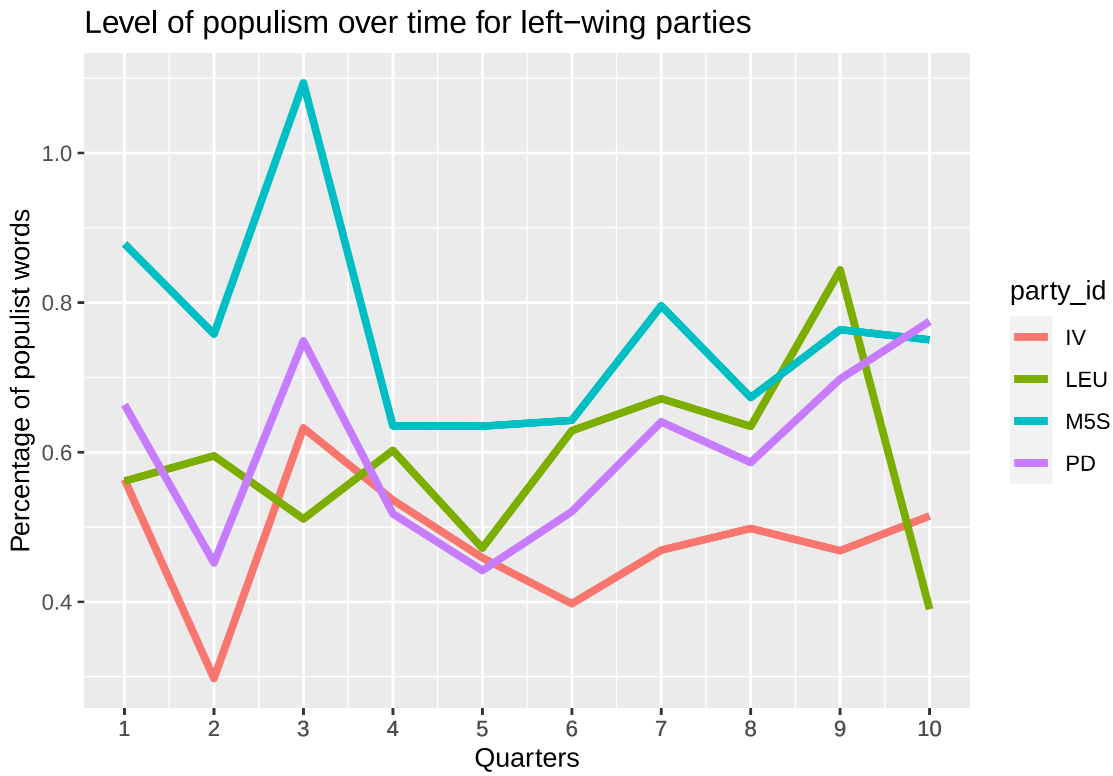
```
##
```

```
## Call:
## lm(formula = populism ~ factor_quarter + factor_party, data = data_dict1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.30617 -0.06571  0.00588  0.05535  0.32599
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  0.60934    0.05058  12.046  < 2e-16 ***
## factor_quarter1              0.04082    0.05058   0.807 0.421838
## factor_quarter2             -0.09418    0.05058  -1.862 0.065878 .
## factor_quarter3              0.13606    0.05058   2.690 0.008522 **
## factor_quarter4             -0.04390    0.05058  -0.868 0.387769
## factor_quarter5             -0.10164    0.05058  -2.009 0.047500 *
## factor_quarter6             -0.07861    0.05058  -1.554 0.123684
## factor_quarter7              0.04596    0.05058   0.909 0.365971
## factor_quarter9              0.09022    0.05058   1.783 0.077879 .
## factor_quarter10            -0.04369    0.05058  -0.864 0.390079
## factor_partyCI              -0.10503    0.05305  -1.980 0.050793 .
## factor_partyFDI              0.19458    0.05305   3.668 0.000414 ***
## factor_partyFI               0.01356    0.05305   0.256 0.798859
## factor_partyINDIPENDENTE -0.14233    0.05305  -2.683 0.008687 **
## factor_partyIV              -0.12078    0.05305  -2.277 0.025184 *
## factor_partyLEGA             0.09147    0.05305   1.724 0.088134 .
## factor_partyLEU             -0.01339    0.05305  -0.252 0.801282
## factor_partyM5S              0.15814    0.05305   2.981 0.003698 **
## factor_partyMISTO            0.03265    0.05305   0.615 0.539799
## factor_partyREG_LEAGUES  -0.18644    0.05305  -3.514 0.000693 ***
```

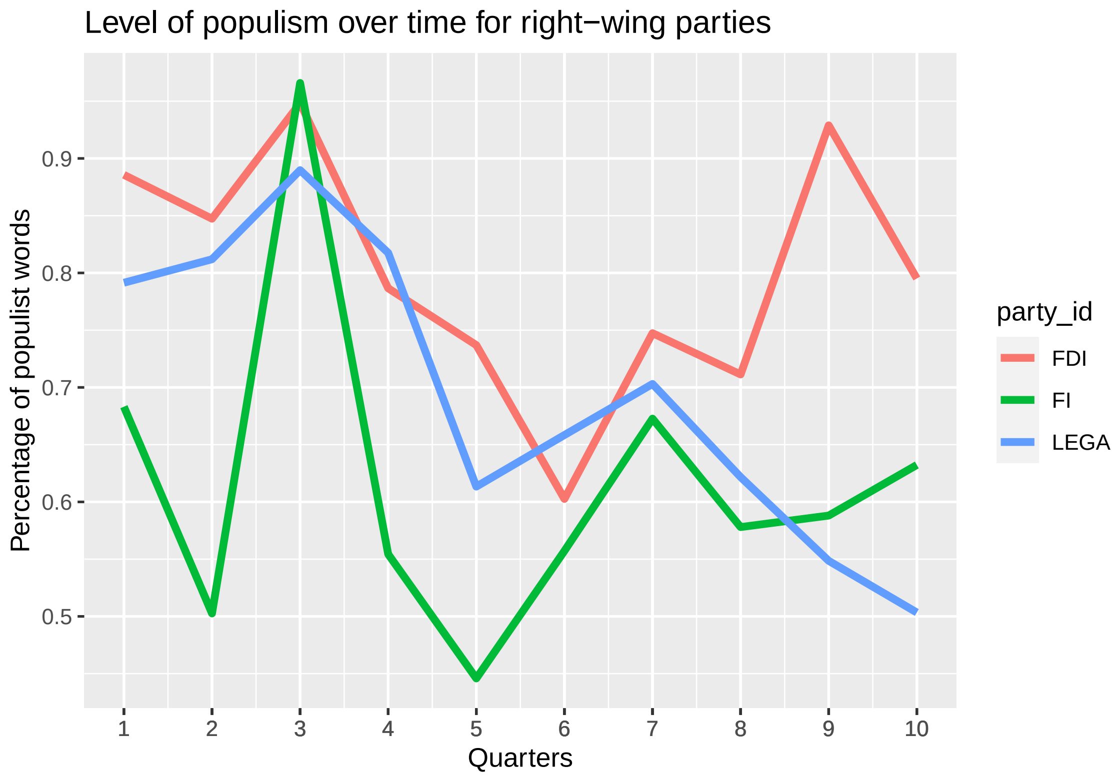
```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1186 on 90 degrees of freedom
## Multiple R-squared:  0.6326, Adjusted R-squared:  0.5551
## F-statistic: 8.157 on 19 and 90 DF,  p-value: 1.35e-12
```

3.2.5 Trends in the level of populism for each parliamentary group over time

```r
#By party & time (quarters)
parties_time <- data_dict1 %>% select(populism, party_id, quarter)

right_party <- data_dict1 %>% select(populism, party_id, quarter) %>%
  filter(party_id == "FDI"|party_id =="FI"|party_id =="LEGA")
left_party <- data_dict1 %>% select(populism, party_id, quarter) %>%
  filter(party_id == "LEU"|party_id =="M5S"|party_id =="PD"|party_id =="IV")

# Left parties in time
ggplot(left_party, aes(x=quarter, y=populism, color=party_id)) +
  geom_line(size=1.5)+
  scale_x_continuous("Quarters", labels = as.character(left_party$quarter), breaks =
  ylab("Percentage of populist words")+
  ggtitle("Level of populism over time for left-wing parties")
```




```r
# Right parties in time
ggplot(right_party, aes(x=quarter, y=populism, color=party_id)) +
  geom_line(size=1.5)+
  scale_x_continuous("Quarters", labels = as.character(right_party$quarter), breaks
  ylab("Percentage of populist words")+
  ggtitle("Level of populism over time for right-wing parties")
```



Level of populism over time for right−wing parties
Percentage of populist words
0.9
0.8
0.7
0.6
0.5
1
2
3
4
5
6
7
8
9
10
Quarters
party_id
FDI
FI
LEGA

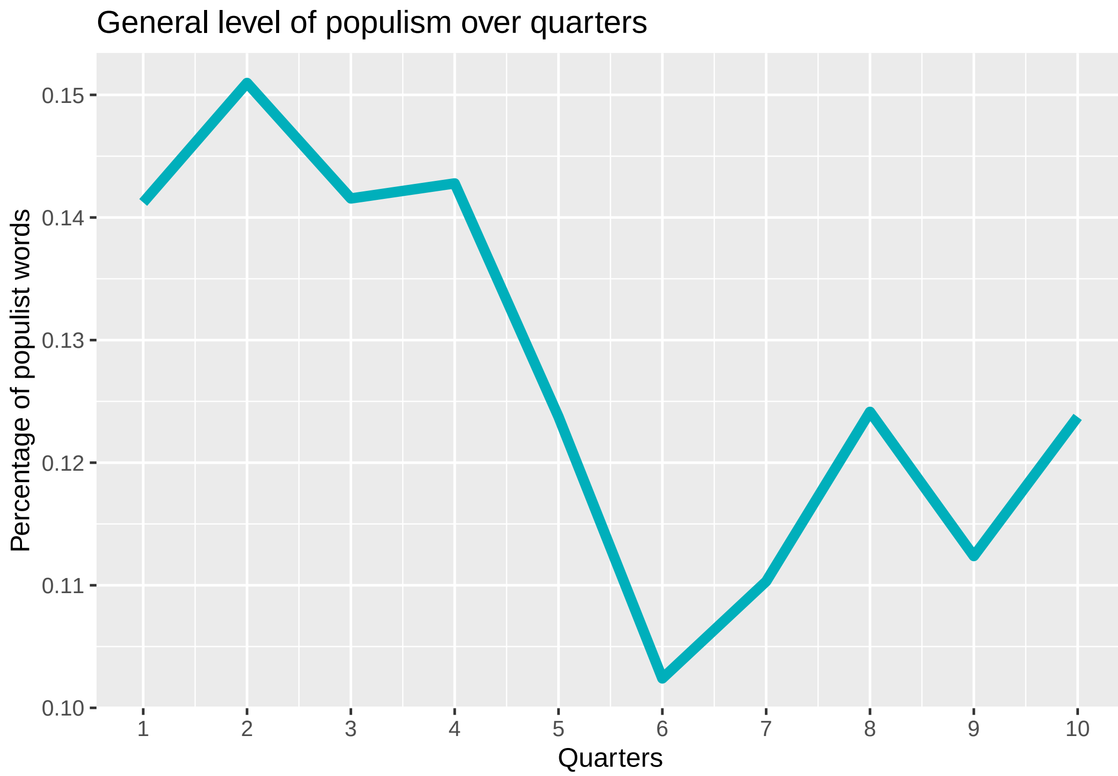
3.3 Rooduijn_Pauwels_Italian

```
# Dictionary analysis with Rooduijn_Pauwels_Italian
# By quarter
dfm_dict2  <- dfm_lookup(dfm_weigh_p_quart, dictionary = Rooduijn_Pauwels_Italian)
```

```
data_dict2 <- dfm_dict2 %>%
  quanteda::convert(to = "data.frame") %>%
  cbind(docvars(dfm_dict2))

# Add variable with general level of populism
#data_dict2 <- data_dict2 %>% mutate(populism = (people + common_will + elite) * 1
```

3.3.1 Level of populism over time

```
#Over time general level populism (quarters)
data_quarter_general2 <- aggregate(x = data_dict2$populism,  # Specify data column
          by = list(data_dict2$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_general2$perc <- data_quarter_general2$x * 100

# plot the level of populism
plot_general2 <- ggplot(data = data_quarter_general2, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_general2$Group.1
  ylab("Percentage of populist words")+
  labs(title = "General level of populism over quarters")
plot_general2
```

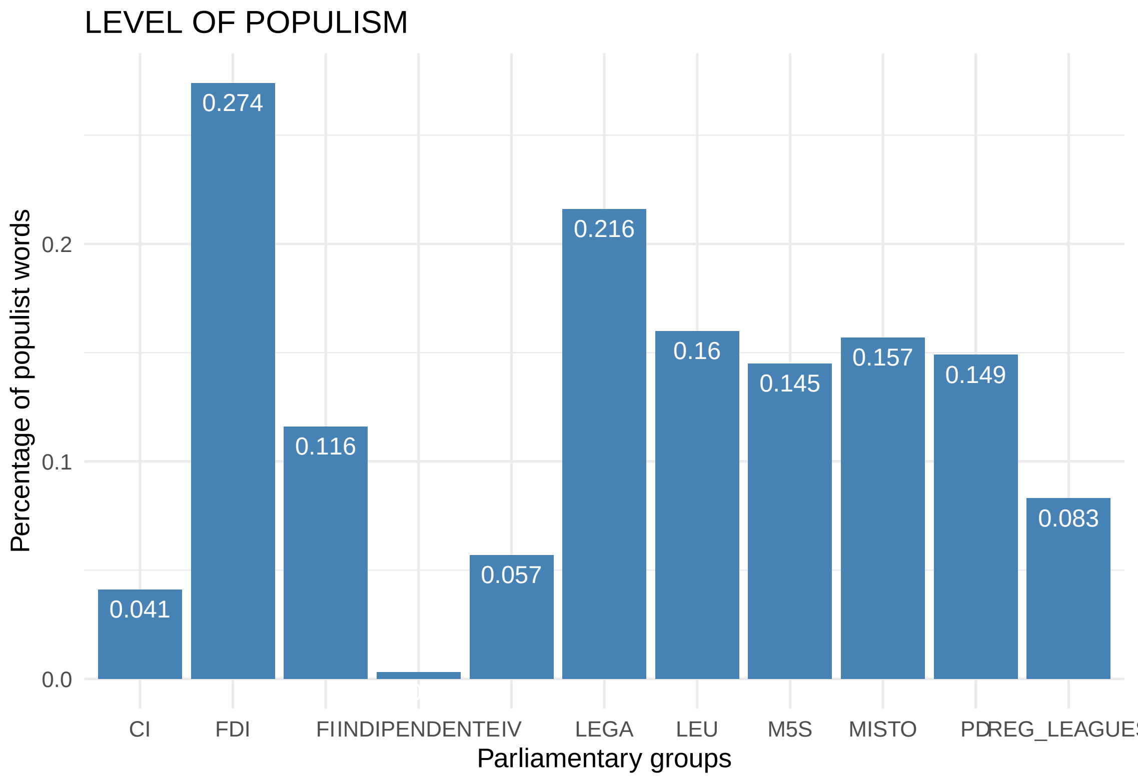


3.3.2 Ranking of parliamentary groups according their populism level

```
# POPULISM
data_party2 <- aggregate(x = data_dict2$populism,  # Specify data column
          by = list(data_dict2$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party2$perc <- round(data_party2$x * 100 ,3)
kable(data_party2 %>% select(Group.1, perc) %>%  arrange(desc(perc)))
```

Group.1	perc
FDI	0.274
LEGA	0.216
LEU	0.160
MISTO	0.157
PD	0.149
M5S	0.145
FI	0.116
REG_LEAGUES	0.083
IV	0.057
CI	0.041
INDIPENDENTE	0.003

```r
ggplot(data=data_party2, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  ylab("Percentage of populist words")+
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM")
```


LEVEL OF POPULISM
Percentage of populist words
0.0
0.1
0.2
0.041
0.274
0.116
0.057
0.216
0.16
0.145
0.157
0.149
0.083
CI
FDI
FI
INDIPENDENTE
IV
LEGA
LEU
M5S
MISTO
PD
REG_LEAGUES
Parliamentary groups

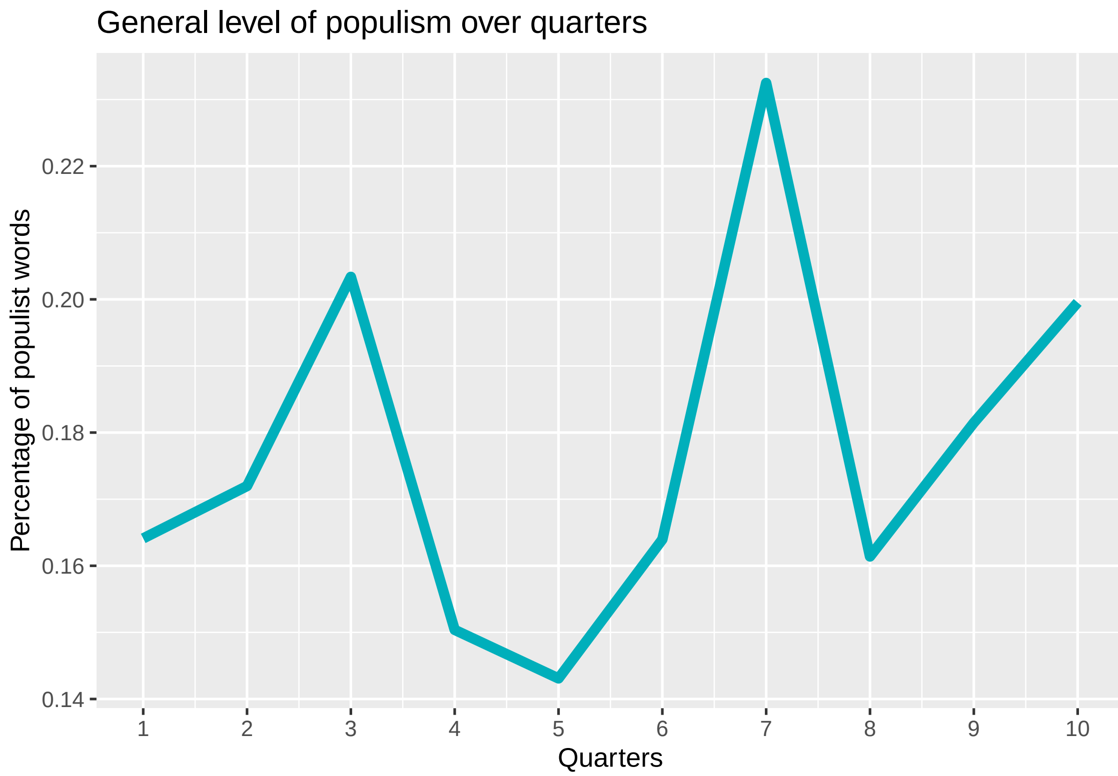
3.4 Grundl_Italian_adapted

```
# Dictionary analysis with Rooduijn_Pauwels_Italian
# By quarter
dfm_dict3  <- dfm_lookup(dfm_weigh_p_quart, dictionary = Grundl_Italian_adapted)
```

```
data_dict3 <- dfm_dict3 %>%
  quanteda::convert(to = "data.frame") %>%
  cbind(docvars(dfm_dict3))

# Add variable with general level of populism
#data_dict2 <- data_dict2 %>% mutate(populism = (people + common_will + elite) * 1
```

3.4.1 Level of populism in time

```
#Over time general level populism (quarters)
data_quarter_general3 <- aggregate(x = data_dict3$populism,  # Specify data column
          by = list(data_dict3$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_general3$perc <- data_quarter_general3$x * 100

# plot the level of populism
plot_general3 <- ggplot(data = data_quarter_general3, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_general3$Group.1
  ylab("Percentage of populist words")+
  labs(title = "General level of populism over quarters")
plot_general3
```

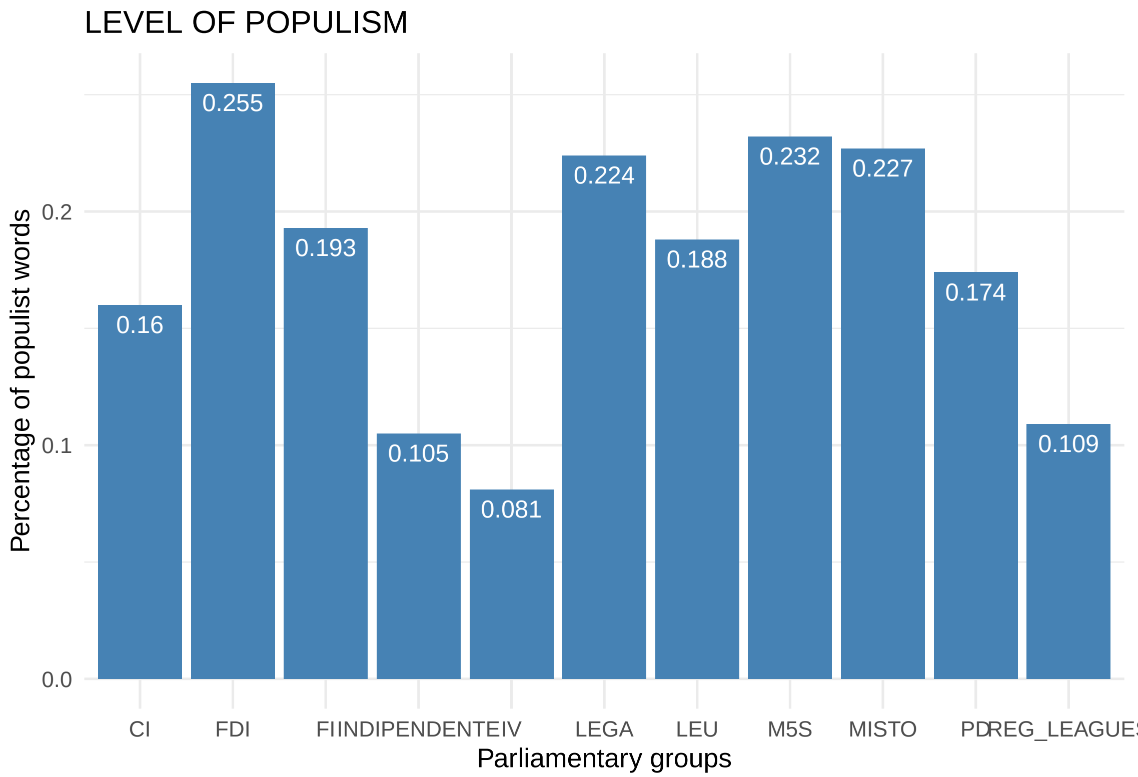


3.4.2 Most populist parliamentary group

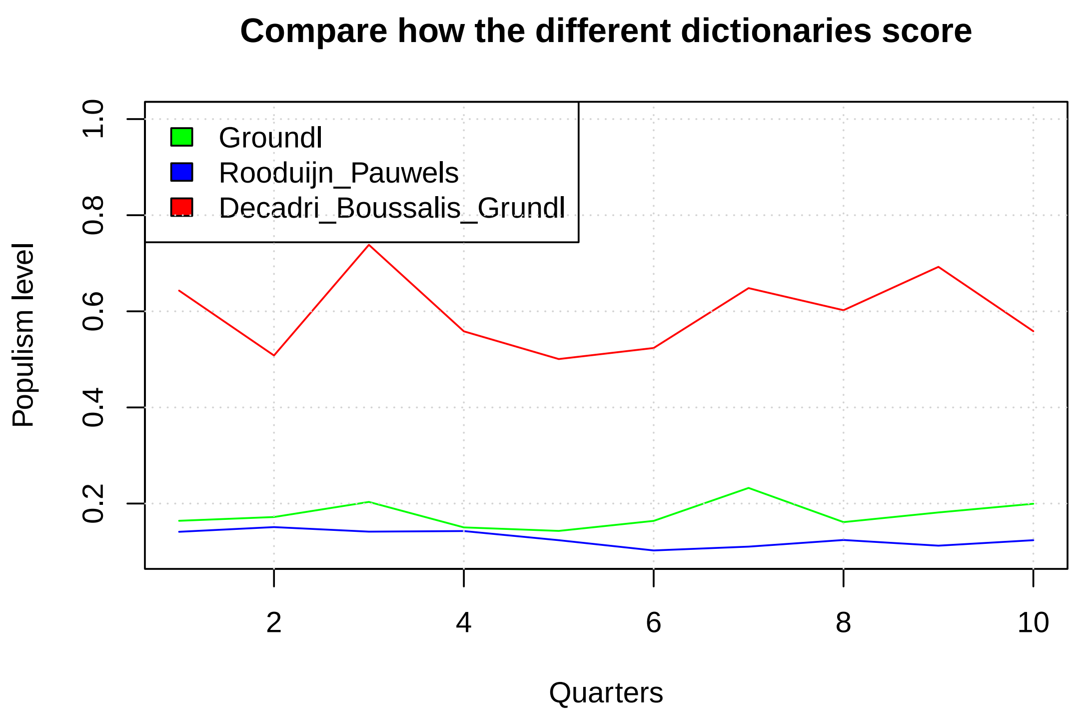
```
# POPULISM
data_party3 <- aggregate(x = data_dict3$populism,  # Specify data column
          by = list(data_dict3$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party3$perc <- round(data_party3$x * 100 ,3)
kable(data_party3 %>% select(Group.1, perc) %>%  arrange(desc(perc)))
```

Group.1	perc
FDI	0.255
M5S	0.232
MISTO	0.227
LEGA	0.224
FI	0.193
LEU	0.188
PD	0.174
CI	0.160
REG_LEAGUES	0.109
INDIPENDENTE	0.105
IV	0.081

```
ggplot(data=data_party3, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=1.6, color="white", size=3.5)+
  theme_minimal()+
  ylab("Percentage of populist words")+
  xlab("Parliamentary groups")+
  labs(title = "LEVEL OF POPULISM")
```


LEVEL OF POPULISM
Percentage of populist words
0.0
0.1
0.2
0.16
0.255
0.193
0.105
0.081
0.224
0.188
0.232
0.227
0.174
0.109
CI
FDI
FI
INDIPENDENTE
IV
LEGA
LEU
M5S
MISTO
PD
REG_LEAGUE
Parliamentary groups

3.5 Compare the general level of populism over time for the dictionaries



3.6 DA SISTEMARE LA COMPARAZIONE TRA DIZIONARI !

3.7 Compare how the dictionaries score for the most populist parliamentary group

```
# Create the columns with the "populist score"
# 11 for the "most populist" and 1 for the least
dfm_dict1_tstat_party_filtered$my_rank <- rank(dfm_dict1_tstat_party_filtered$popul
dfm_dict2_tstat_party$my_rank <- rank(dfm_dict2_tstat_party$frequency)
```

```
dict_3_tstat_party$my_rank <- rank(dict_3_tstat_party$frequency)
dict_4_tstat_party$my_rank <- rank(dict_4_tstat_party$frequency)

# define the parliamentary group list
party <- c("LEGA","PD","M5S","FI","FDI","MISTO",
           "LEU","CI","IV","INDIPENDENTE","REG_LEAGUES")
# create an empty df
party_rank <- data.frame(first = vector(), second = vector(),
                         third = vector(), fourth = vector(), fifth = vector() )

# loop the rank for each parliamentary group
for (i in party)
{
  rank_dict_1 <- (dfm_dict1_tstat_party_filtered %>% filter(group == i ) %>% .$my_ra
  rank_dict_2 <- (dfm_dict2_tstat_party %>% filter(group == i ) %>% .$my_rank)
  rank_dict_3 <- (dict_3_tstat_party %>% filter(group == i ) %>% .$my_rank)
  rank_dict_4 <- (dict_4_tstat_party %>% filter(group == i ) %>% .$my_rank)

  party <- (i)
  party_rank <- rbind(party_rank, cbind(party, rank_dict_1, rank_dict_2,
                                        rank_dict_3, rank_dict_4))
}

# change the format of the columns in numeric
party_rank$rank_dict_1 <- as.numeric(party_rank$rank_dict_1)
party_rank$rank_dict_2 <- as.numeric(party_rank$rank_dict_2)
party_rank$rank_dict_3 <- as.numeric(party_rank$rank_dict_3)
party_rank$rank_dict_4 <- as.numeric(party_rank$rank_dict_4)
```

```r
# Create the column with the sum of the single score
party_rank$total_score <- rowSums(party_rank[,-1])
kable(party_rank %>% arrange(desc(total_score)), col.names = c("Party",
                                                               "Dec_Bous_Grun",
                                                               "Rood_Pau_it",
                                                               "Grun_it",
                                                               "Dec_Bous",
                                                               "Total"))
```

```r
# Data
load("data/dfm.Rda")

# Dictionary LWIC Complete
LWIC_ITA <- dictionary(file = "data/large_files/Italian_LIWC2007_Dictionary.dic",
                       format = "LIWC")
```

```
## note: removing empty key: Formale
```

```
## note: removing empty key: Passivo
```

```r
emotions <- c("Emo_Pos", "Emo_Neg", "Ansia", "Rabbia", "Tristezza", "Ottimismo" )

# Count the number of words
n.words <- c(
length(LWIC_ITA[["Emo_Pos"]]),
length(LWIC_ITA[["Emo_Neg"]]),
length(LWIC_ITA[["Ansia"]]),
length(LWIC_ITA[["Rabbia"]]),
length(LWIC_ITA[["Tristez"]]),
length(LWIC_ITA[["Ottimis"]])
)

num_words <-  data.frame(emotions,n.words)

# Extracting only the keys we need
myLWIC_ITA <- dictionary(list(positive = LWIC_ITA[["Emo_Pos"]],
                              negative = LWIC_ITA[["Emo_Neg"]],
                              anxiety = LWIC_ITA[["Ansia"]],
```

```
                                     anger = LWIC_ITA[["Rabbia"]],
                                     sadness = LWIC_ITA[["Tristez"]],
                                     optimism = LWIC_ITA[["Ottimis"]]))

myLWIC_ITA_sent <- dictionary(list(positive = LWIC_ITA[["Emo_Pos"]],
                                     negative = LWIC_ITA[["Emo_Neg"]]))
```

```
kable(num_words)
```

emotions	n.words
Emo_Pos	200
Emo_Neg	663
Ansia	65
Rabbia	227
Tristezza	226
Ottimismo	93

3.7.1 Group and weight the dfm

```
# By party & quarter
dfm_weigh_p_quart <- dfm_group(DFM, groups = interaction(party_id, quarter))%>%
  dfm_weight(scheme = "prop")
```

3.8 Apply the dictionary

```
# Apply Dictionary to DFM
DFM_emotions <- dfm_lookup(dfm_weigh_p_quart,
```

```r
                            dictionary = myLWIC_ITA)
DFM_emotions
```

```
## Document-feature matrix of: 110 documents, 6 features (0.76% sparse) and 3 docva
##                   features
## docs                  positive   negative    anxiety       anger     sadness
##   CI.1           0.008060854 0.02236603 0.003405995 0.006471390 0.004541326
##   FDI.1          0.006416312 0.02893245 0.002834199 0.011061250 0.006140765
##   FI.1           0.006498830 0.02547256 0.003243474 0.007675035 0.006974064
##   INDIPENDENTE.1 0.005129667 0.01567398 0.001994870 0.005984611 0.003989741
##   IV.1           0.008545455 0.02309091 0.003272727 0.009272727 0.006000000
##   LEGA.1         0.006352373 0.02593448 0.003005565 0.008426081 0.006194876
##                   features
## docs                 optimism
##   CI.1           0.01089918
##   FDI.1          0.01487955
##   FI.1           0.01447089
##   INDIPENDENTE.1 0.01025933
##   IV.1           0.01600000
##   LEGA.1         0.01257350
## [ reached max_ndoc ... 104 more documents ]
```

3.8.1 Transform the DFM into an ordinary dataframe

```r
data_dict_emo <- DFM_emotions %>%
  quanteda::convert(to = "data.frame") %>%
  cbind(docvars(DFM_emotions))
```

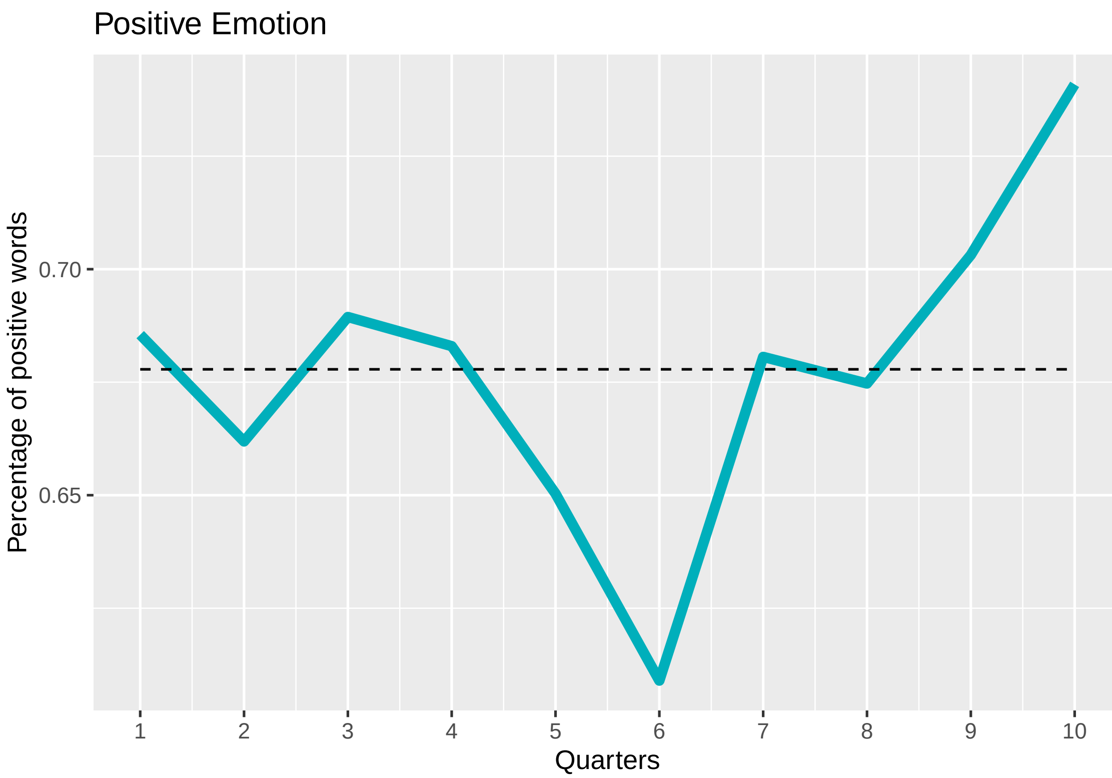
```
# Create a new variable with the difference of negative - positive emotions
data_dict_emo$negative_prevalence <- (data_dict_emo$negative - data_dict_emo$positi

# Transform the proportion into percentage
data_dict_emo <- data_dict_emo %>% mutate(positive =  positive * 100,
                                          negative = negative * 100,
                                          anxiety = anxiety * 100,
                                          anger = anger *100,
                                          sadness = sadness * 100,
                                          optimism = optimism *100,
                                          negative_prevalence = negative_prevalence
```

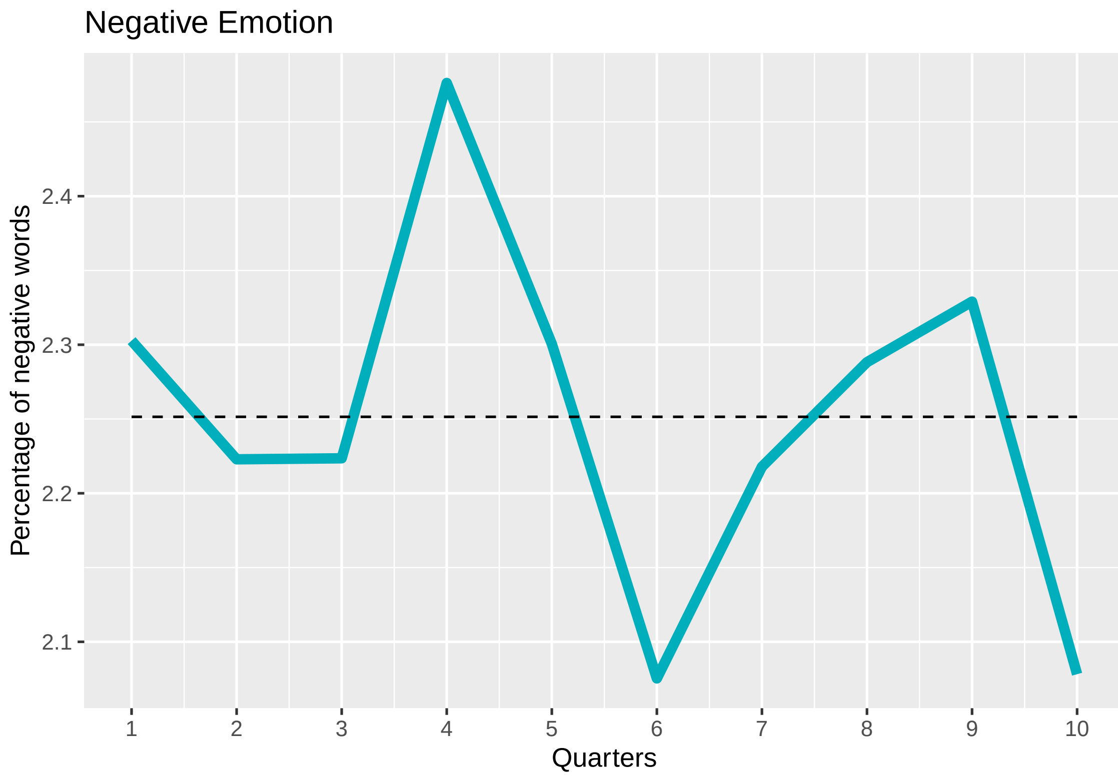
3.9 Percentage of the emotions in time

The code is only shown for 'positive' but is identical for all emotions

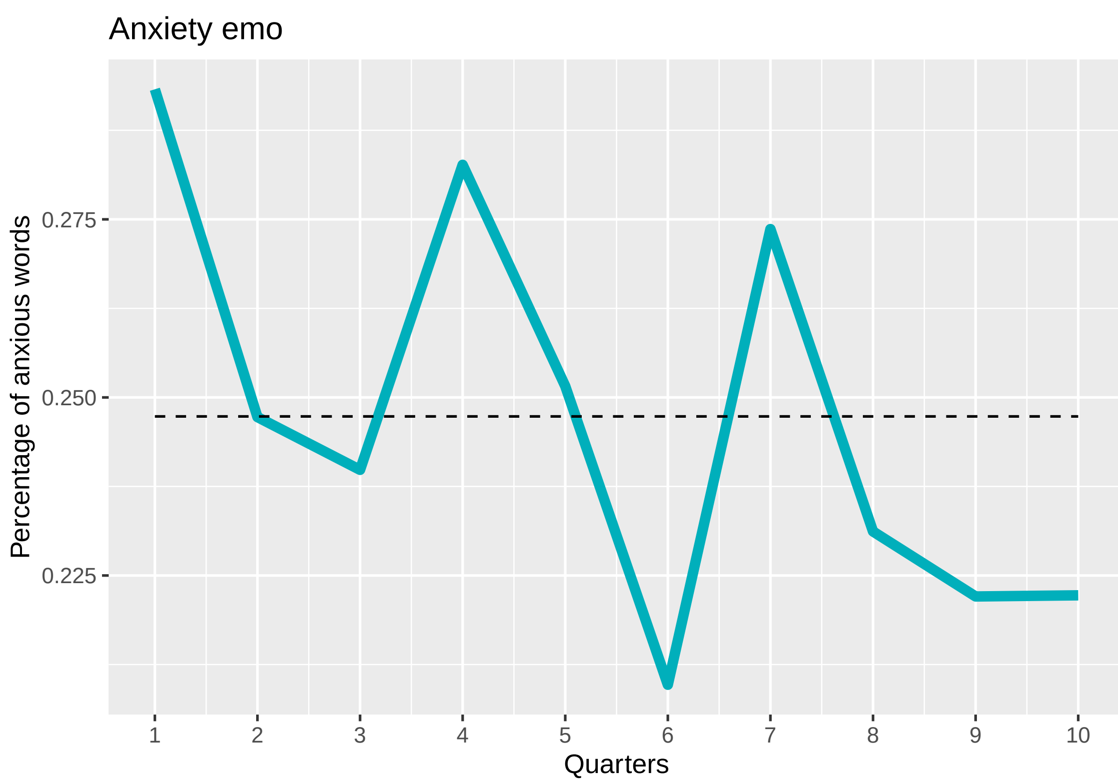
```
#Over time POSITIVE (quarters)
data_quarter_positive <- aggregate(x = data_dict_emo$positive,  # Specify data colu
          by = list(data_dict_emo$quarter),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_quarter_positive$perc <- data_quarter_positive$x

mean_positive <- mean(data_quarter_positive$perc)

# plot
plot_positive <- ggplot(data = data_quarter_positive, aes(x = Group.1, y = perc))+
  geom_line(color = "#00AFBB", size = 2)+
  scale_x_continuous("Quarters", labels = as.character(data_quarter_positive$Group.
  geom_line(aes(x = Group.1, y = mean(perc)),linetype = "dashed")+
```

```
  ylab("Percentage of positive words")+
  labs(title = "Positive Emotion")
plot_positive
```

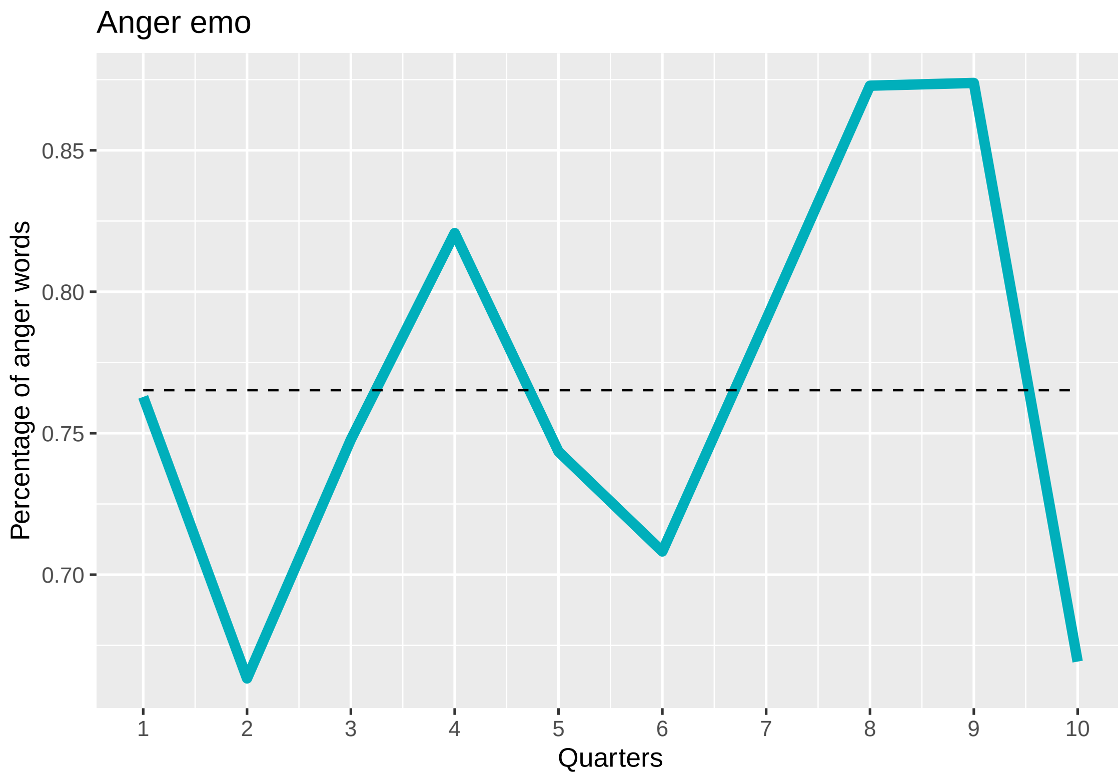


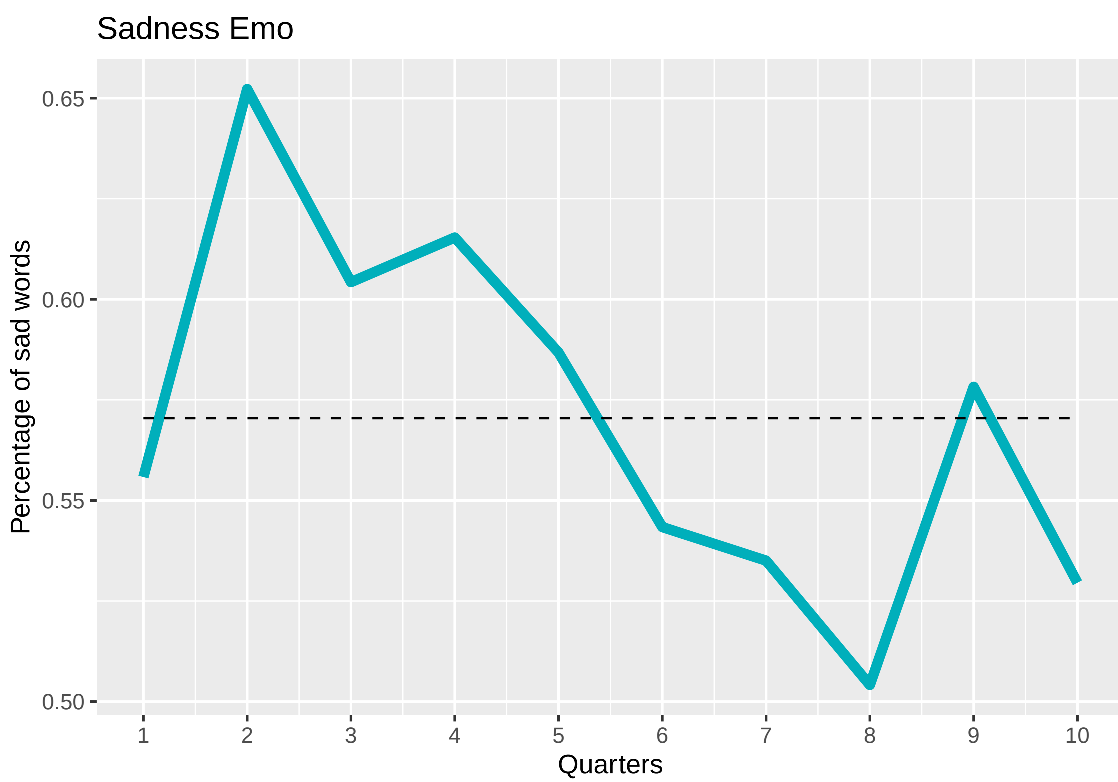


Negative Emotion
Percentage of negative words
2.4
2.3
2.2
2.1
1
2
3
4
5
6
7
8
9
10
Quarters

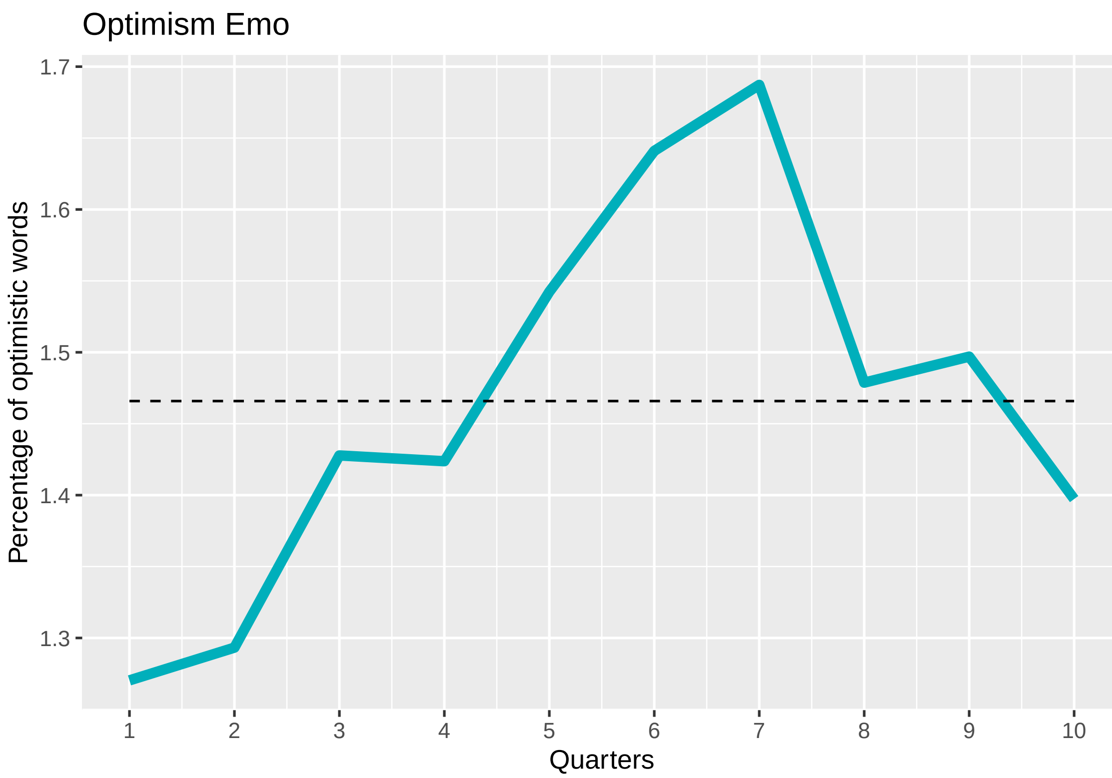


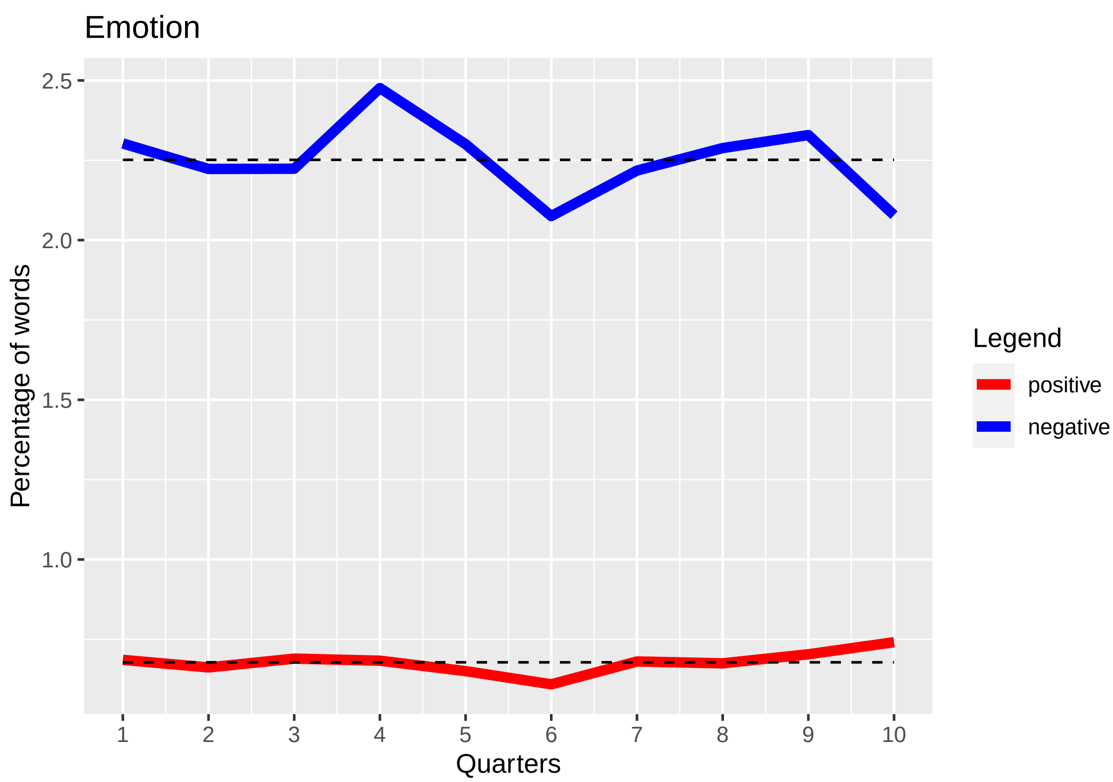
Anxiety emo
Percentage of anxious words
0.275
0.250
0.225
1
2
3
4
5
6
7
8
9
10
Quarters

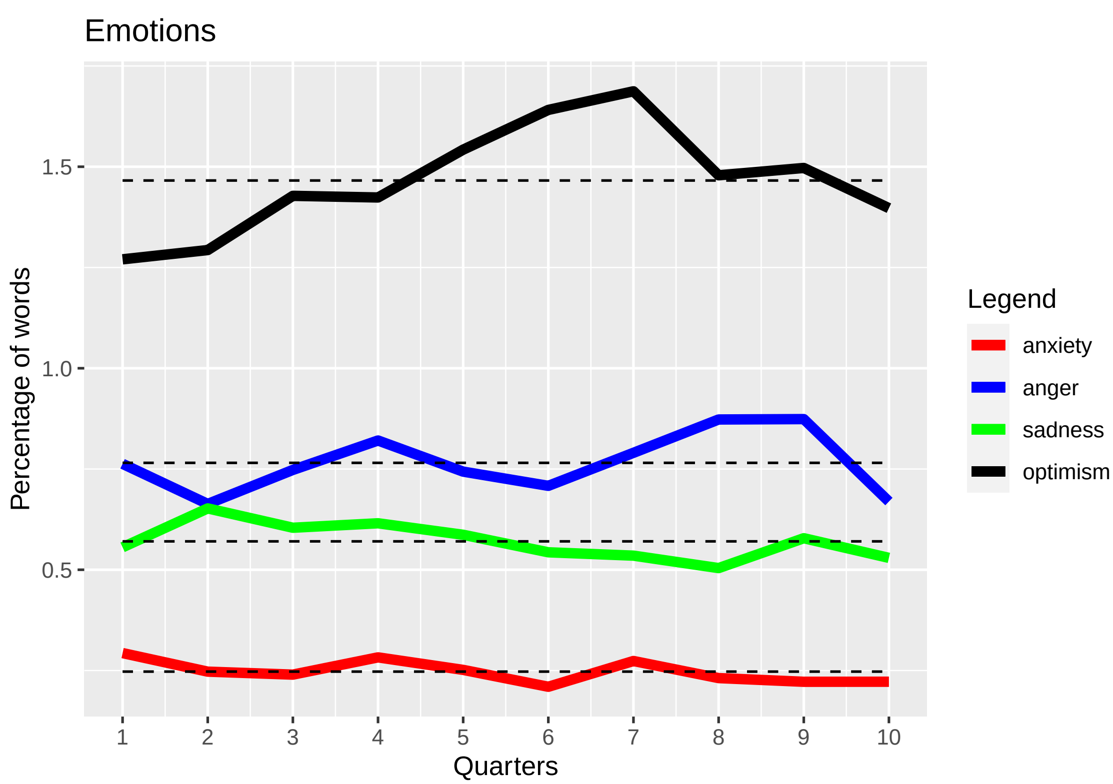

Anger emo
Percentage of anger words
0.85
0.80
0.75
0.70
1
2
3
4
5
6
7
8
9
10
Quarters


Sadness Emo
Percentage of sad words
0.65
0.60
0.55
0.50
1
2
3
4
5
6
7
8
9
10
Quarters

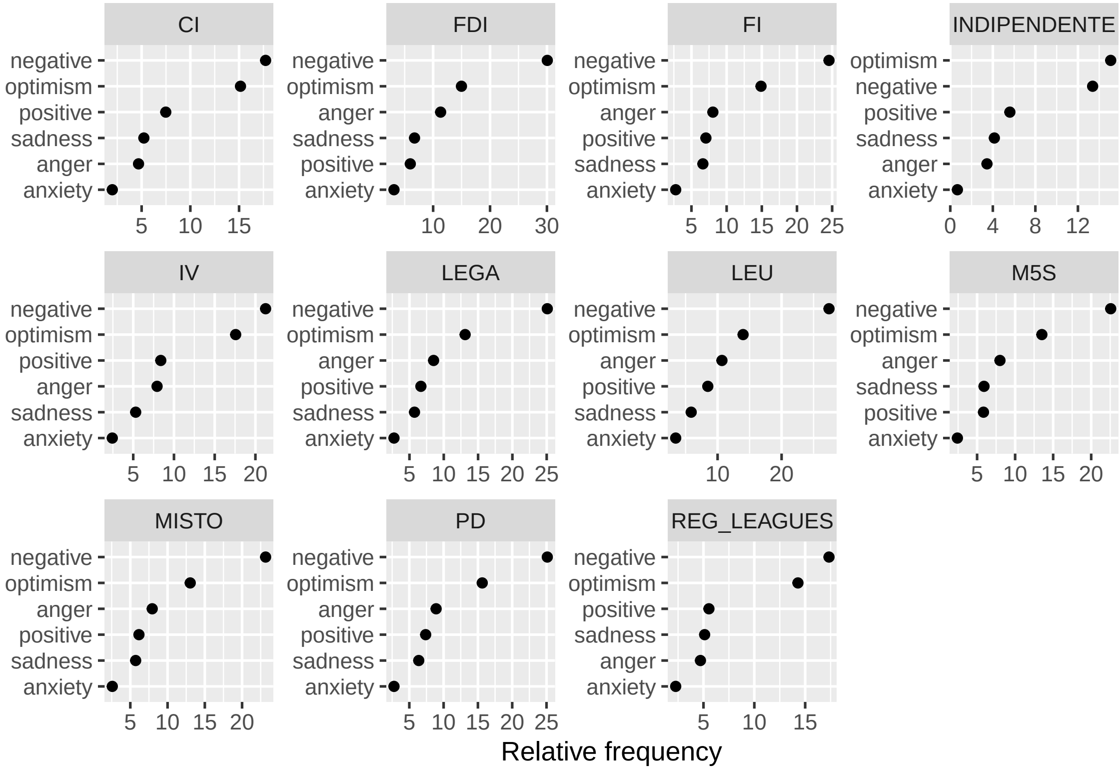
```
## function (x, y, ...)
## UseMethod("plot")
## <bytecode: 0x000001c1df5f87b0>
## <environment: namespace:base>
```




Emotion
Percentage of words
2.5
2.0
1.5
1.0
1
2
3
4
5
6
7
8
9
10
Quarters
Legend
positive
negative


Emotions
Percentage of words
1.5
1.0
0.5
1
2
3
4
5
6
7
8
9
10
Quarters
Legend
anxiety
anger
sadness
optimism

3.10 Main emotion for each parliamentary group



The code is only shown for 'positive' but is identical for all emotions

```
# POSITIVE
data_party_positive <- aggregate(x = data_dict_emo$positive,  # Specify data column
          by = list(data_dict_emo$party_id),  # Specify group indicator
          FUN = mean) # Specify function (i.e. mean)
data_party_positive$perc <- round(data_party_positive$x,3)
kable(data_party_positive %>% select(Group.1, perc) %>%  arrange(desc(perc)), captio
```

```
ggplot(data=data_party_positive, aes(x=Group.1, y=perc)) +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=perc), vjust=0, color="black", size=3.5)+
  geom_abline(slope=0, intercept= mean(data_party_positive$perc),lty=2) +
```

Table 1: POSITIVE

Group.1	perc
LEU	0.847
IV	0.838
CI	0.748
PD	0.738
FI	0.706
LEGA	0.667
MISTO	0.616
FDI	0.598
M5S	0.584
INDIPENDENTE	0.560
REG_LEAGUES	0.554

```
theme_minimal()+
xlab("Parliamentary group")+
labs(title = "Positive Emotion")+
coord_flip()
```

Table 2: NEGATIVE

Group.1	perc
FDI	3.006
LEU	2.741
PD	2.512
LEGA	2.509
FI	2.455
MISTO	2.316
M5S	2.257
IV	2.125
CI	1.772
REG_LEAGUES	1.734
INDIPENDENTE	1.338

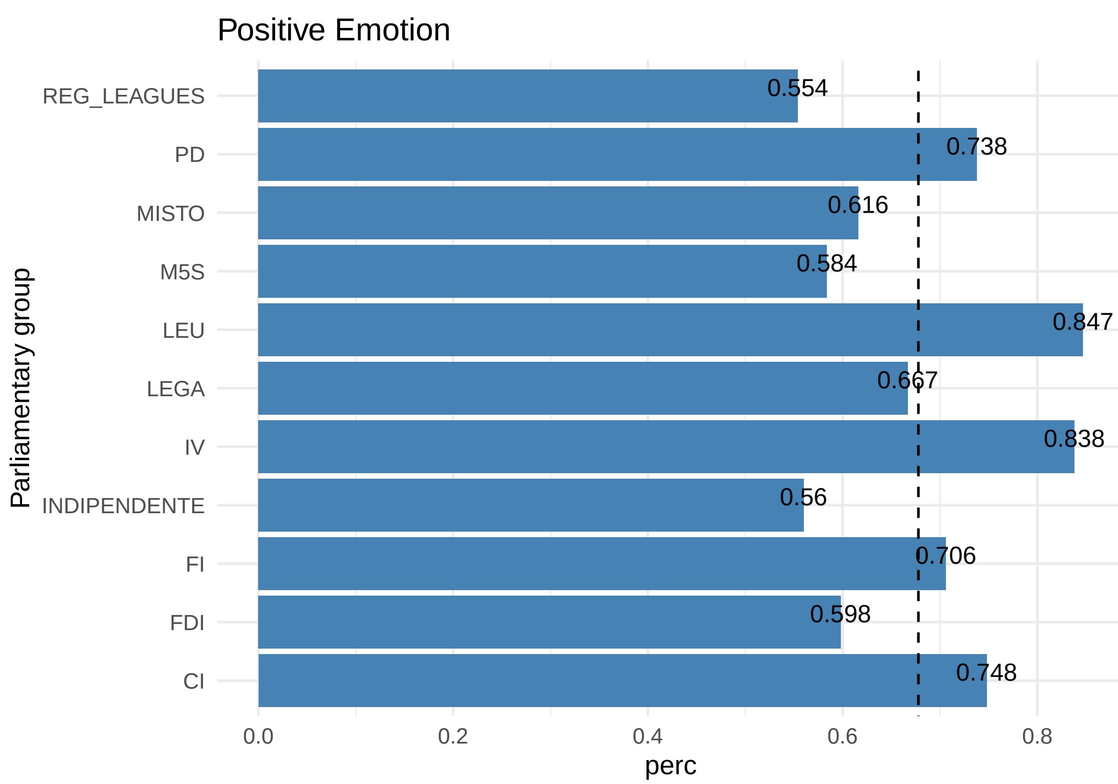


Table 3: ANXIETY

Group.1	perc
LEU	0.345
FDI	0.312
PD	0.277
FI	0.276
LEGA	0.275
MISTO	0.258
IV	0.243
M5S	0.241
REG_LEAGUES	0.227
CI	0.199
INDIPENDENTE	0.067

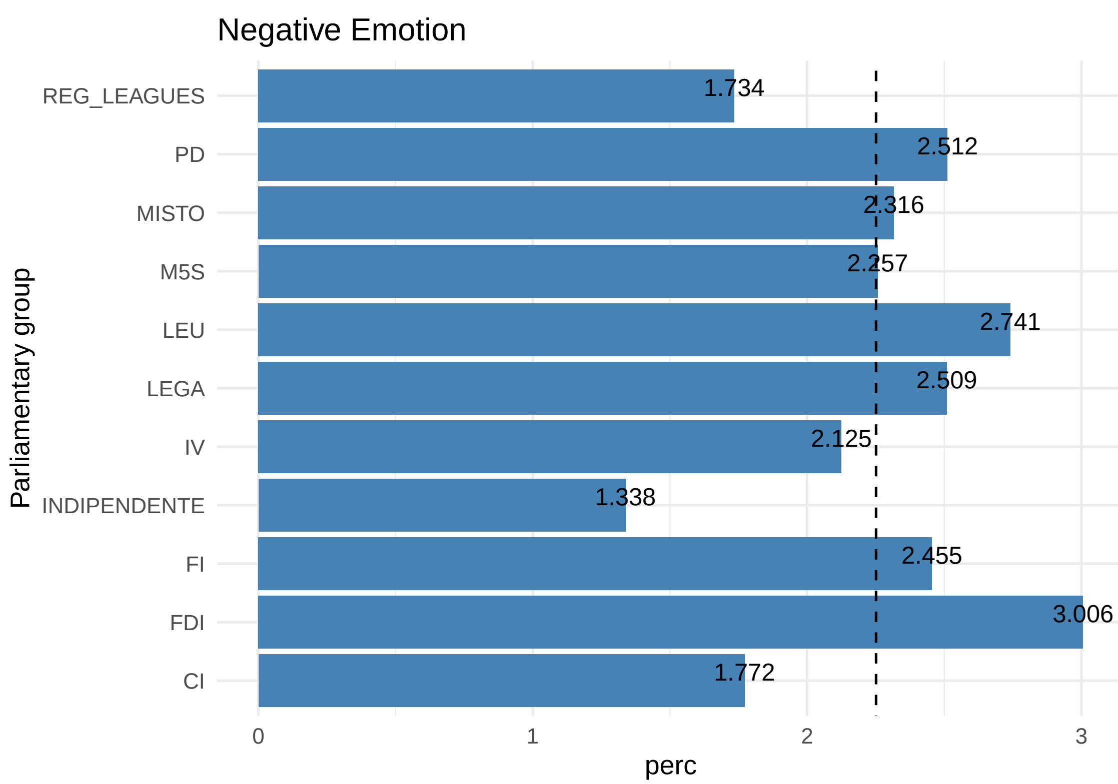


Table 4: ANGER

Group.1	perc
FDI	1.132
LEU	1.068
PD	0.891
LEGA	0.852
FI	0.805
M5S	0.801
MISTO	0.794
IV	0.793
REG_LEAGUES	0.470
CI	0.468
INDIPENDENTE	0.345

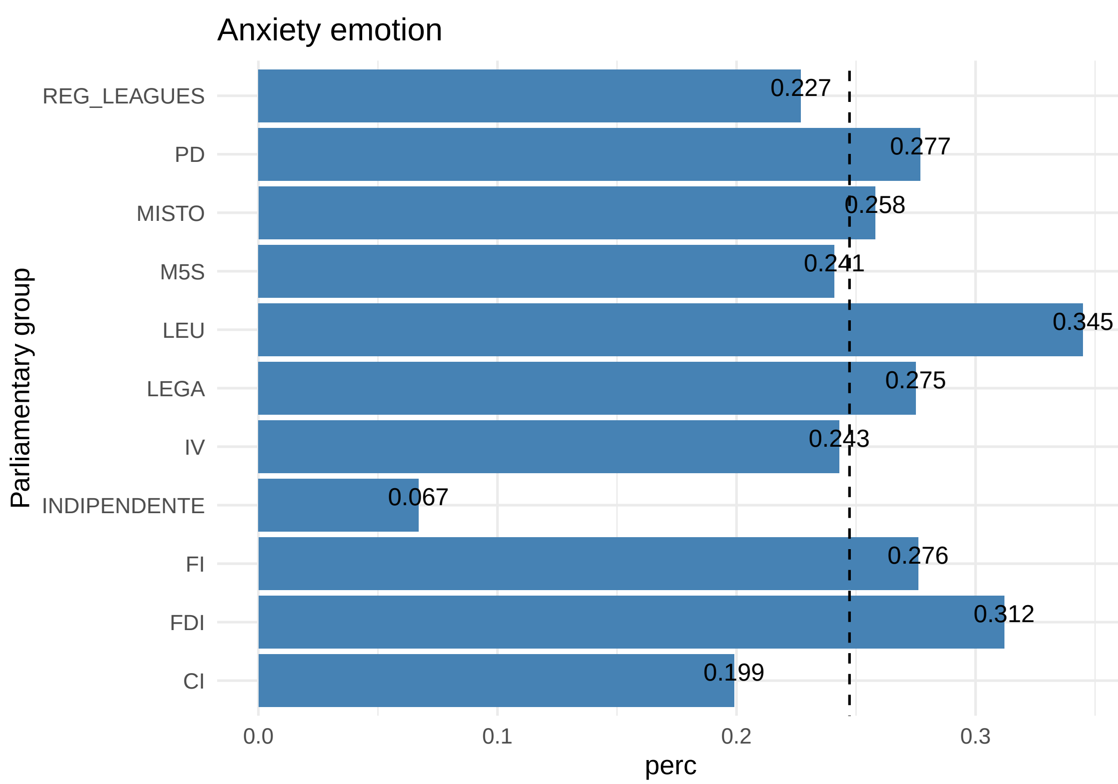


Table 5: SADNESS

Group.1	perc
FDI	0.673
FI	0.663
PD	0.638
M5S	0.591
LEU	0.587
LEGA	0.573
MISTO	0.572
IV	0.530
CI	0.523
REG_LEAGUES	0.511
INDIPENDENTE	0.414

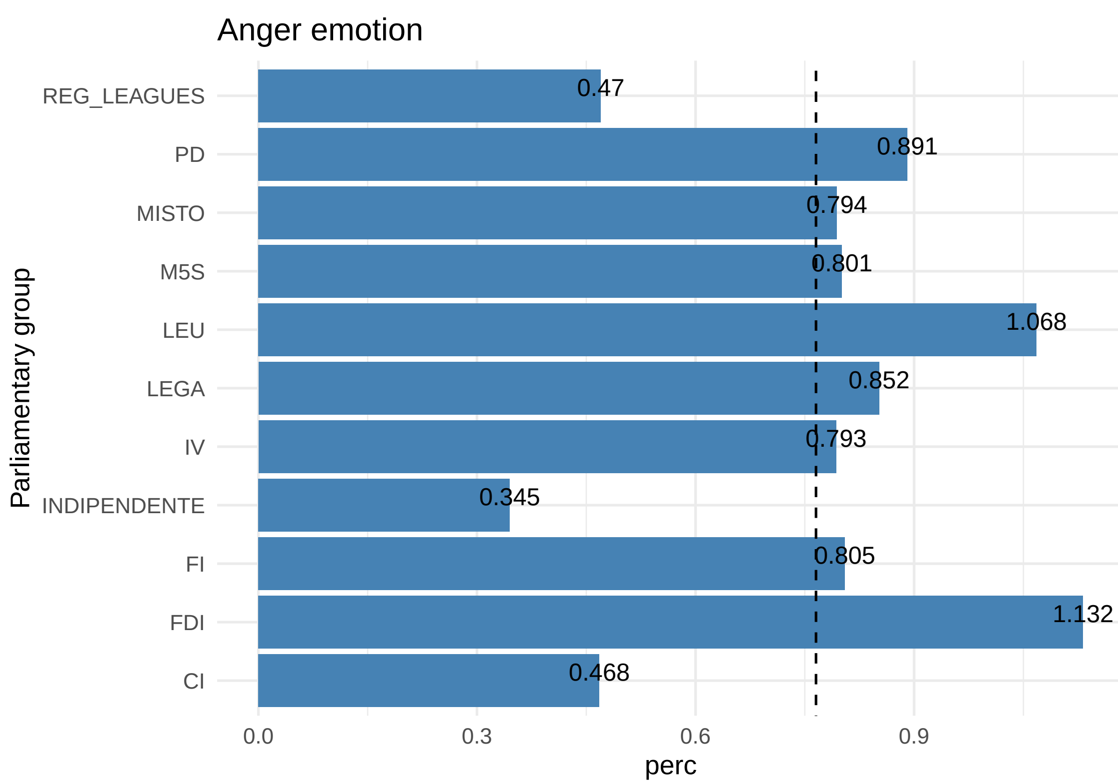
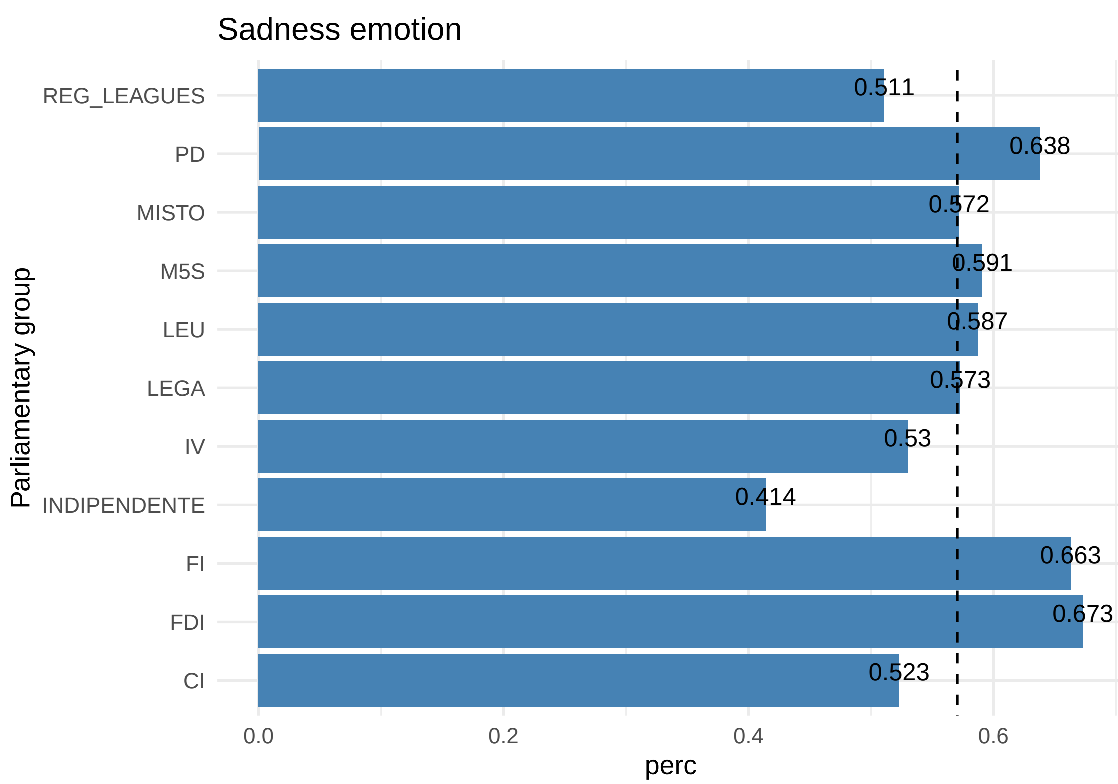
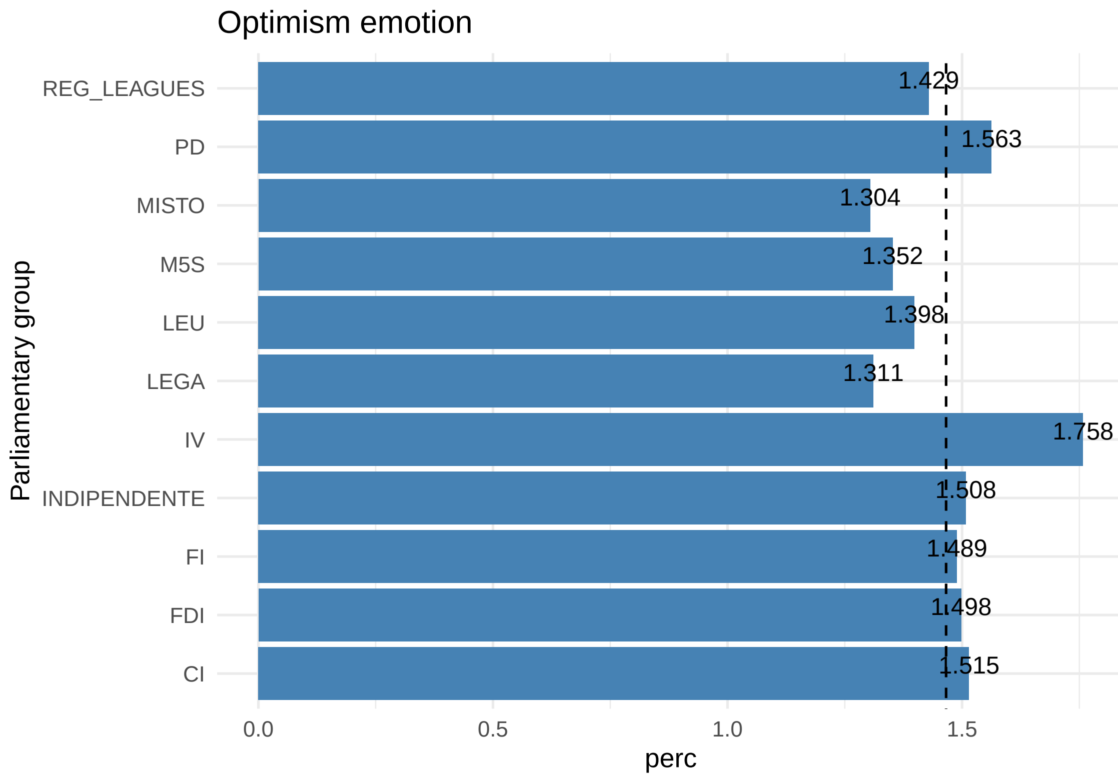


Table 6: OPTIMISM

Group.1	perc
IV	1.758
PD	1.563
CI	1.515
INDIPENDENTE	1.508
FDI	1.498
FI	1.489
REG_LEAGUES	1.429
LEU	1.398
M5S	1.352
LEGA	1.311
MISTO	1.304





3.10.1 Are the average values of positive/negative emotions for each party statistically different from each other?

The reference category is PD

```
# bivariate regression for check t-test

# create the factor variables for party and quarter
data_dict_emo$factor_party <- as.factor(data_dict_emo$party_id)
data_dict_emo$factor_quarter <- as.factor(data_dict_emo$quarter)

# Check the mean values
summary(data_dict_emo$positive)
```

```
##     Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
## 0.3281 0.5863 0.6542 0.6778 0.7546 1.1593
```

```
summary(data_dict_emo$negative)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.9522  1.9364  2.3318  2.2515  2.5867  3.2025
```

```
# Set PD as reference category for party_id
data_dict_emo$factor_party <- relevel(data_dict_emo$factor_party, ref = "PD")
# Set 5 as reference category for quarter
data_dict_emo$factor_quarter <- relevel(data_dict_emo$factor_quarter, ref = "5")

# Run the regressions

# POSITIVE
positive_model <- lm(positive ~  factor_quarter + factor_party, data_dict_emo )
summary(positive_model)
```

```
##
## Call:
## lm(formula = positive ~ factor_quarter + factor_party, data = data_dict_emo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.26194 -0.06684  0.00093  0.04680  0.33861
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 0.710990   0.052210  13.618  < 2e-16 ***
```

```
## factor_quarter1           0.035165   0.052210   0.674  0.50234
## factor_quarter2           0.011541   0.052210   0.221  0.82556
## factor_quarter3           0.039079   0.052210   0.748  0.45611
## factor_quarter4           0.032630   0.052210   0.625  0.53358
## factor_quarter6          -0.041367   0.052210  -0.792  0.43026
## factor_quarter7           0.030252   0.052210   0.579  0.56376
## factor_quarter8           0.024362   0.052210   0.467  0.64191
## factor_quarter9           0.052797   0.052210   1.011  0.31462
## factor_quarter10          0.090541   0.052210   1.734  0.08632 .
## factor_partyCI            0.009462   0.054759   0.173  0.86321
## factor_partyFDI          -0.140003   0.054759  -2.557  0.01224 *
## factor_partyFI           -0.032835   0.054759  -0.600  0.55026
## factor_partyINDIPENDENTE -0.178239   0.054759  -3.255  0.00160 **
## factor_partyIV            0.099436   0.054759   1.816  0.07272 .
## factor_partyLEGA         -0.071907   0.054759  -1.313  0.19247
## factor_partyLEU           0.108649   0.054759   1.984  0.05029 .
## factor_partyM5S          -0.154273   0.054759  -2.817  0.00595 **
## factor_partyMISTO        -0.122489   0.054759  -2.237  0.02776 *
## factor_partyREG_LEAGUES  -0.184902   0.054759  -3.377  0.00109 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1224 on 90 degrees of freedom
## Multiple R-squared:  0.4781, Adjusted R-squared:  0.3679
## F-statistic: 4.339 on 19 and 90 DF,  p-value: 1.009e-06
```

```
#NEGATIVE
negative_model <- lm(negative ~  factor_quarter + factor_party, data_dict_emo )
summary(negative_model)
```

```
##
## Call:
## lm(formula = negative ~ factor_quarter + factor_party, data = data_dict_emo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.79357 -0.14849  0.00431  0.15790  0.46872
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              2.560662   0.108714  23.554  < 2e-16 ***
## factor_quarter1          0.002167   0.108714   0.020  0.98414
## factor_quarter2         -0.077716   0.108714  -0.715  0.47654
## factor_quarter3         -0.077039   0.108714  -0.709  0.48038
## factor_quarter4          0.175647   0.108714   1.616  0.10966
## factor_quarter6         -0.225225   0.108714  -2.072  0.04115 *
## factor_quarter7         -0.082757   0.108714  -0.761  0.44851
## factor_quarter8         -0.012345   0.108714  -0.114  0.90984
## factor_quarter9          0.028457   0.108714   0.262  0.79410
## factor_quarter10        -0.222362   0.108714  -2.045  0.04374 *
## factor_partyCI          -0.739253   0.114020  -6.484 4.70e-09 ***
## factor_partyFDI          0.494954   0.114020   4.341 3.71e-05 ***
## factor_partyFI          -0.056139   0.114020  -0.492  0.62366
## factor_partyINDIPENDENTE -1.173282   0.114020 -10.290  < 2e-16 ***
## factor_partyIV          -0.386425   0.114020  -3.389  0.00104 **
## factor_partyLEGA        -0.002478   0.114020  -0.022  0.98271
## factor_partyLEU          0.229343   0.114020   2.011  0.04727 *
## factor_partyM5S         -0.254663   0.114020  -2.233  0.02800 *
## factor_partyMISTO       -0.195756   0.114020  -1.717  0.08944 .
```

```
## factor_partyREG_LEAGUES  -0.777217   0.114020  -6.817 1.03e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.255 on 90 degrees of freedom
## Multiple R-squared:  0.8089, Adjusted R-squared:  0.7685
## F-statistic: 20.05 on 19 and 90 DF,  p-value: < 2.2e-16
```

3.11 Regressions

```
# import the populism dataset
load("data/data_dict1.Rda")

# add the level of populism in the dataframe with the emotions
data_dict_emo$populism <- data_dict1$populism

# Change the reference category for quarter as quarter 8
data_dict_emo$factor_quarter <- relevel(data_dict_emo$factor_quarter, ref = "8")

# Negative prevalence
negative_prevalence_model <- lm(negative_prevalence ~ factor_party + factor_quarter
summary(negative_prevalence_model)
```

```
##
## Call:
## lm(formula = negative_prevalence ~ factor_party + factor_quarter +
##     populism, data = data_dict_emo)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.83425 -0.13061 -0.01836  0.15555  0.69102
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1.457921   0.196396   7.423 6.51e-11 ***
## factor_partyCI            -0.687517   0.130189  -5.281 9.02e-07 ***
## factor_partyFDI            0.521583   0.136636   3.817 0.000249 ***
## factor_partyFI            -0.031204   0.127490  -0.245 0.807208
## factor_partyINDIPENDENTE -0.912110   0.132441  -6.887 7.79e-10 ***
## factor_partyIV            -0.415488   0.131061  -3.170 0.002090 **
## factor_partyLEGA           0.016135   0.129531   0.125 0.901148
## factor_partyLEU            0.128497   0.127488   1.008 0.316228
## factor_partyM5S           -0.192532   0.133586  -1.441 0.153021
## factor_partyMISTO         -0.092293   0.127711  -0.723 0.471778
## factor_partyREG_LEAGUES   -0.483682   0.135906  -3.559 0.000600 ***
## factor_quarter5            0.095929   0.124208   0.772 0.441968
## factor_quarter1           -0.020075   0.121951  -0.165 0.869623
## factor_quarter2            0.002328   0.123831   0.019 0.985041
## factor_quarter3           -0.158689   0.126302  -1.256 0.212250
## factor_quarter4            0.205304   0.122020   1.683 0.095969 .
## factor_quarter6           -0.101347   0.123132  -0.823 0.412663
## factor_quarter7           -0.103082   0.122068  -0.844 0.400675
## factor_quarter9           -0.040199   0.123641  -0.325 0.745849
## factor_quarter10          -0.250742   0.122015  -2.055 0.042810 *
## populism                   0.582670   0.253212   2.301 0.023721 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.285 on 89 degrees of freedom
## Multiple R-squared:  0.7629, Adjusted R-squared:  0.7096
## F-statistic: 14.32 on 20 and 89 DF,  p-value: < 2.2e-16
```

```
# Negative emotion
negative_model <- lm(negative ~ factor_party + factor_quarter + populism, data_dict_
summary(negative_model)
```

```
##
## Call:
## lm(formula = negative ~ factor_party + factor_quarter + populism,
##     data = data_dict_emo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.82801 -0.13125  0.00941  0.12134  0.50310
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 2.248269   0.171994  13.072  < 2e-16 ***
## factor_partyCI             -0.687535   0.114013  -6.030 3.65e-08 ***
## factor_partyFDI             0.399141   0.119659   3.336  0.00124 **
## factor_partyFI             -0.062815   0.111649  -0.563  0.57511
## factor_partyINDIPENDENTE -1.103196   0.115985  -9.511 3.28e-15 ***
## factor_partyIV             -0.326952   0.114777  -2.849  0.00545 **
## factor_partyLEGA           -0.047517   0.113437  -0.419  0.67631
## factor_partyLEU             0.235937   0.111648   2.113  0.03738 *
## factor_partyM5S            -0.332532   0.116988  -2.842  0.00555 **
```

```
## factor_partyMISTO        -0.211835   0.111843  -1.894  0.06147 .
## factor_partyREG_LEAGUES  -0.685412   0.119020  -5.759 1.19e-07 ***
## factor_quarter5           0.062394   0.108775   0.574  0.56768
## factor_quarter1          -0.005587   0.106799  -0.052  0.95840
## factor_quarter2          -0.018994   0.108445  -0.175  0.86136
## factor_quarter3          -0.131691   0.110609  -1.191  0.23698
## factor_quarter4           0.209609   0.106859   1.962  0.05294 .
## factor_quarter6          -0.174171   0.107833  -1.615  0.10981
## factor_quarter7          -0.093044   0.106902  -0.870  0.38644
## factor_quarter9          -0.003622   0.108279  -0.033  0.97339
## factor_quarter10         -0.188505   0.106855  -1.764  0.08114 .
## populism                  0.492414   0.221751   2.221  0.02892 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2496 on 89 degrees of freedom
## Multiple R-squared:  0.8189, Adjusted R-squared:  0.7782
## F-statistic: 20.12 on 20 and 89 DF,  p-value: < 2.2e-16
```

```
# Anxiety emotion
anxiety_model <- lm(anxiety ~ factor_party + factor_quarter + populism, data_dict_em
summary(anxiety_model)
```

```
##
## Call:
## lm(formula = anxiety ~ factor_party + factor_quarter + populism,
##     data = data_dict_emo)
##
## Residuals:
```

```
##       Min        1Q    Median        3Q       Max
## -0.203185 -0.030062 -0.006422  0.031150  0.241173
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)               0.2688373  0.0478034   5.624 2.13e-07 ***
## factor_partyCI           -0.0792116  0.0316883  -2.500   0.0143 *
## factor_partyFDI           0.0378298  0.0332575   1.137   0.2584
## factor_partyFI           -0.0006212  0.0310313  -0.020   0.9841
## factor_partyINDIPENDENTE -0.2119155  0.0322366  -6.574 3.24e-09 ***
## factor_partyIV           -0.0357351  0.0319007  -1.120   0.2656
## factor_partyLEGA         -0.0010955  0.0315281  -0.035   0.9724
## factor_partyLEU           0.0681484  0.0310310   2.196   0.0307 *
## factor_partyM5S          -0.0338173  0.0325152  -1.040   0.3011
## factor_partyMISTO        -0.0182670  0.0310853  -0.588   0.5583
## factor_partyREG_LEAGUES  -0.0526060  0.0330799  -1.590   0.1153
## factor_quarter5           0.0190702  0.0302326   0.631   0.5298
## factor_quarter1           0.0626135  0.0296833   2.109   0.0377 *
## factor_quarter2           0.0148207  0.0301407   0.492   0.6241
## factor_quarter3           0.0104310  0.0307423   0.339   0.7352
## factor_quarter4           0.0509013  0.0297000   1.714   0.0900 .
## factor_quarter6          -0.0225554  0.0299707  -0.753   0.4537
## factor_quarter7           0.0430576  0.0297118   1.449   0.1508
## factor_quarter9          -0.0079431  0.0300946  -0.264   0.7924
## factor_quarter10         -0.0095388  0.0296988  -0.321   0.7488
## populism                 -0.0131192  0.0616326  -0.213   0.8319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 0.06936 on 89 degrees of freedom
## Multiple R-squared:  0.5817, Adjusted R-squared:  0.4877
## F-statistic: 6.188 on 20 and 89 DF,  p-value: 6.176e-10
```

```r
# Anger emotion
anger_model <- lm(anger ~ factor_party + factor_quarter + populism, data_dict_emo)
summary(anger_model)
```

```
##
## Call:
## lm(formula = anger ~ factor_party + factor_quarter + populism,
##     data = data_dict_emo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32401 -0.07952  0.00037  0.06871  0.48334
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 0.88129    0.09360   9.415 5.19e-15 ***
## factor_partyCI             -0.40239    0.06205  -6.485 4.83e-09 ***
## factor_partyFDI             0.20315    0.06512   3.120  0.00244 **
## factor_partyFI             -0.08894    0.06076  -1.464  0.14678
## factor_partyINDIPENDENTE -0.51858    0.06312  -8.215 1.57e-12 ***
## factor_partyIV             -0.07514    0.06246  -1.203  0.23221
## factor_partyLEGA           -0.05692    0.06174  -0.922  0.35900
## factor_partyLEU             0.17934    0.06076   2.951  0.00404 **
## factor_partyM5S            -0.12072    0.06367  -1.896  0.06120 .
## factor_partyMISTO          -0.10324    0.06087  -1.696  0.09337 .
```

```
## factor_partyREG_LEAGUES  -0.38502    0.06477  -5.944 5.33e-08 ***
## factor_quarter5          -0.10977    0.05920  -1.854  0.06701 .
## factor_quarter1          -0.11785    0.05812  -2.028  0.04559 *
## factor_quarter2          -0.19139    0.05902  -3.243  0.00167 **
## factor_quarter3          -0.15128    0.06020  -2.513  0.01377 *
## factor_quarter4          -0.04364    0.05816  -0.750  0.45502
## factor_quarter6          -0.14951    0.05869  -2.548  0.01256 *
## factor_quarter7          -0.09150    0.05818  -1.573  0.11934
## factor_quarter9          -0.01639    0.05893  -0.278  0.78149
## factor_quarter10         -0.19516    0.05815  -3.356  0.00116 **
## populism                  0.19253    0.12068   1.595  0.11418
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1358 on 89 degrees of freedom
## Multiple R-squared:  0.8022, Adjusted R-squared:  0.7577
## F-statistic: 18.04 on 20 and 89 DF,  p-value: < 2.2e-16
```

```
# sadness emotion
sadness_model <- lm(sadness ~ factor_party + factor_quarter + populism, data_dict_em
summary(sadness_model)
```

```
##
## Call:
## lm(formula = sadness ~ factor_party + factor_quarter + populism,
##     data = data_dict_emo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.36628 -0.04760  0.00219  0.04560  0.36965
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               0.51962    0.08025   6.475 5.06e-09 ***
## factor_partyCI           -0.10570    0.05320  -1.987 0.049995 *
## factor_partyFDI           0.01902    0.05583   0.341 0.734222
## factor_partyFI            0.02387    0.05209   0.458 0.647930
## factor_partyINDIPENDENTE -0.21123    0.05412  -3.903 0.000184 ***
## factor_partyIV           -0.09736    0.05355  -1.818 0.072438 .
## factor_partyLEGA         -0.07186    0.05293  -1.358 0.178028
## factor_partyLEU          -0.04913    0.05209  -0.943 0.348193
## factor_partyM5S          -0.06025    0.05459  -1.104 0.272693
## factor_partyMISTO        -0.06847    0.05219  -1.312 0.192868
## factor_partyREG_LEAGUES  -0.11049    0.05553  -1.990 0.049710 *
## factor_quarter5           0.09126    0.05075   1.798 0.075556 .
## factor_quarter1           0.04824    0.04983   0.968 0.335682
## factor_quarter2           0.15611    0.05060   3.085 0.002710 **
## factor_quarter3           0.08862    0.05161   1.717 0.089436 .
## factor_quarter4           0.11495    0.04986   2.306 0.023463 *
## factor_quarter6           0.04591    0.05031   0.912 0.363979
## factor_quarter7           0.02701    0.04988   0.542 0.589448
## factor_quarter9           0.06648    0.05052   1.316 0.191568
## factor_quarter10          0.02911    0.04986   0.584 0.560799
## populism                  0.08471    0.10347   0.819 0.415138
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1164 on 89 degrees of freedom
```

```
## Multiple R-squared:  0.3902, Adjusted R-squared:  0.2532
## F-statistic: 2.847 on 20 and 89 DF,  p-value: 0.0003978
```

3.11.1 Tab the results

```
tab_model(negative_model,negative_prevalence_model)
```

negative

negative_prevalence

Predictors

Estimates

CI

p

Estimates

CI

p

(Intercept)

2.25

1.91 – 2.59

<0.001

1.46

1.07 – 1.85

<0.001

factor party [CI]

-0.69

-0.91 – -0.46

<0.001

-0.69

-0.95 – -0.43

<0.001

factor party [FDI]

0.40

0.16 – 0.64

0.001

0.52

0.25 – 0.79

<0.001

factor party [FI]

-0.06

-0.28 – 0.16

0.575

-0.03

-0.28 – 0.22

0.807

factor party[INDIPENDENTE]

-1.10

-1.33 – -0.87

<0.001

-0.91

-1.18 – -0.65

<0.001

factor party [IV]

-0.33

-0.56 – -0.10

0.005

-0.42

-0.68 – -0.16

0.002

factor party [LEGA]

-0.05

-0.27 – 0.18

0.676

0.02

-0.24 – 0.27

0.901

factor party [LEU]

0.24

0.01 – 0.46

0.037

0.13

-0.12 – 0.38

0.316

factor party [M5S]

-0.33

-0.56 – -0.10

0.006

-0.19

-0.46 – 0.07

0.153

factor party [MISTO]

-0.21

-0.43 – 0.01

0.061

-0.09

-0.35 – 0.16

0.472

factor party[REG_LEAGUES]

-0.69

-0.92 – -0.45

<0.001

-0.48

-0.75 – -0.21

0.001

factor quarter [5]

0.06

-0.15 – 0.28

0.568

0.10

-0.15 – 0.34

0.442

factor quarter [1]

-0.01

-0.22 – 0.21

0.958

-0.02

-0.26 – 0.22

0.870

factor quarter [2]

-0.02

-0.23 – 0.20

0.861

0.00

-0.24 – 0.25

0.985

factor quarter [3]

-0.13

-0.35 – 0.09

0.237

-0.16

-0.41 – 0.09

0.212

factor quarter [4]

0.21

-0.00 – 0.42

0.053

0.21

-0.04 – 0.45

0.096

factor quarter [6]

-0.17

-0.39 – 0.04

0.110

-0.10

-0.35 – 0.14

0.413

factor quarter [7]

-0.09

-0.31 – 0.12

0.386

-0.10

-0.35 – 0.14

0.401

factor quarter [9]

-0.00

-0.22 – 0.21

0.973

-0.04

-0.29 – 0.21

0.746

factor quarter [10]

-0.19

-0.40 – 0.02

0.081

-0.25

-0.49 – -0.01

0.043

populism

0.49

0.05 – 0.93

0.029

0.58

0.08 – 1.09

0.024

Observations

110

110

R2 / R2 adjusted

0.819 / 0.778

0.763 / 0.710

4 STM Topic model analysis

4.1 Prelimnary steps

4.1.1 Load the data

4.1.2 Import the dictionaries

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

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)]))
```

4.1.3 Remove all the account's mentions

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

4.1.4 Trim the data

```
# Remove text with less than 1 word
DFM <- dfm_subset(DFM, ntoken(DFM) > 1)

# Remove very short words
DFM <- dfm_remove(DFM, min_nchar=2)

# Dfm trimming:only words that occur in the top 20% of the distribution and in les
DFM <- dfm_trim(DFM,min_termfreq = 0.8,
                termfreq_type = "quantile",
                max_docfreq = 0.1, docfreq_type = "prop")
```

4.1.5 Group and weight the data

```
DFMG<-dfm_group(DFM,groups = interaction(nome, quarter,party_id))

DFMGW<-  dfm_weight(DFMG,
                    scheme ="prop")
```

4.1.6 Apply dictionary

```
# Apply Dictionary
DFMdict <- dfm_lookup(DFMGW, dictionary = Decadri_Boussalis_Grundl)

# Convert to a dataframe
DATAdictDFM <- DFMdict %>%
  quanteda::convert(to = "data.frame")
```

4.1.7 Create percentage for each components

```
# RUN ONLY ONCE!
# Add variable with general level of populism & multiply all components by 100
DATAdictDFM <- DATAdictDFM %>% mutate(populism = (people + common_will + elite) * 10

DATAdictDFM <- DATAdictDFM %>% mutate(people = people*100,
                                       common_will = common_will*100,
                                       elite = elite*100)
```

4.1.8 Add the percentage of populism to the original dfm (not weighted)

```
docvars(DFMG) <- cbind(docvars(DFMG),DATAdictDFM)
```

4.1.9 Convert DFM to STM format

```
#EXtract a sample
#myDFM <- dfm_sample(DFMG,
#                     size = 5000)

myDFM = DFMG
set.seed(123)
DfmStm <- quanteda::convert(myDFM, to = "stm", docvars = docvars(myDFM))
```

```
## Warning in dfm2stm(x, docvars, omit_empty = TRUE): Dropped empty document(s): DE
## LORENZIS Diego.7.M5S, DI MICCO Fabio.8.M5S, MARZANA Maria.9.M5S
```

4.2 FIND THE BEST NUMBER OF TOPICS K

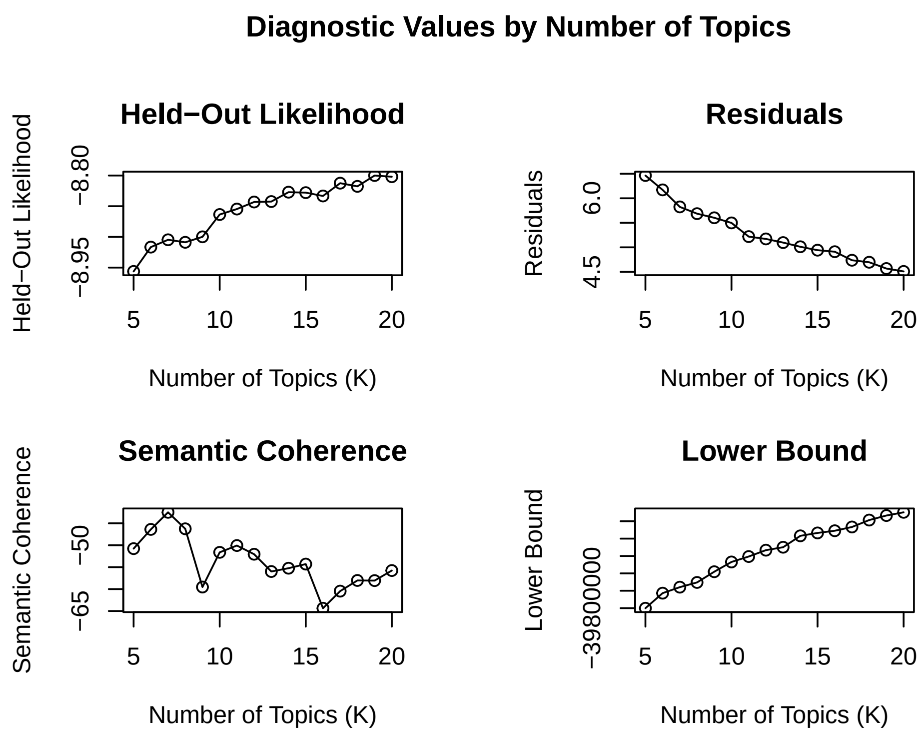
4.2.1 Search the best number of Topics comparing coherence and exclusivity values

K = 2:50

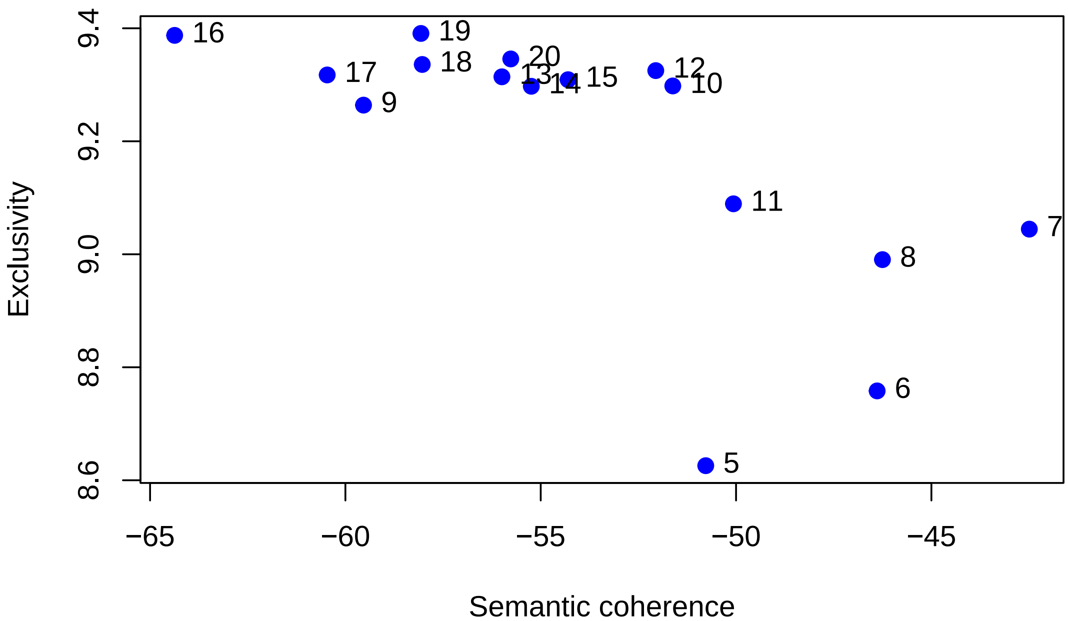
```
k <-c(5:20)
system.time(storage <- searchK(DfmStm$documents,
                               DfmStm$vocab,
                               K = k,
                               #max.em.its = 75, # REMOVED
                               prevalence = ~ party_id + populism + s(quarter),
                               data = DfmStm$meta, init.type = "Spectral"))
#save(storage,file="data/storage.Rda")
```

4.2.2 plot results

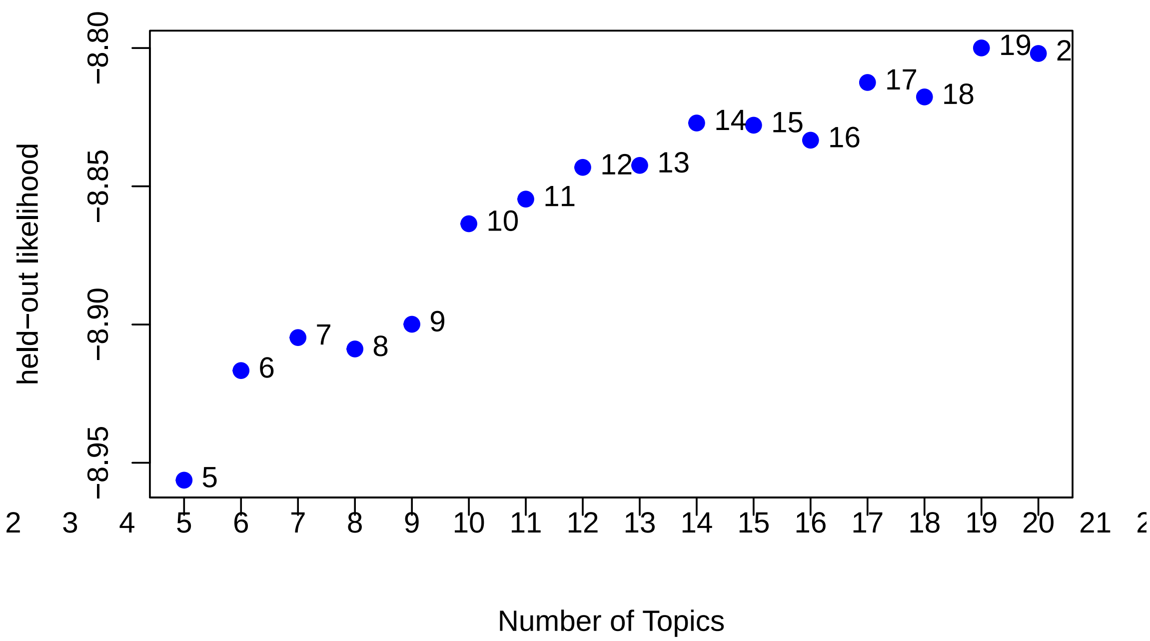
```
plot.searchK(storage)
```




```
plot(storage$results$semcoh, storage$results$exclus,
     xlab= "Semantic coherence",
     ylab= "Exclusivity",
     col= "blue", pch = 19, cex = 1, lty = "solid", lwd = 2)
text(storage$results$semcoh, storage$results$exclus, labels=storage$results$K, cex=
```




```
plot(storage$results$K, storage$results$heldout,
     xlab= "Number of Topics",
     ylab= "held-out likelihood",
     col= "blue", pch = 19, cex = 1, lty = "solid", lwd = 2, xaxt="n")
text(storage$results$K, storage$results$heldout, labels=storage$results$K, cex= 1, p
xtick<-seq(2, 50, by=1)
axis(side=1, at=xtick, labels = FALSE)
text(x=xtick,  par("usr")[3],
     labels = xtick, pos = 1, xpd = TRUE)
```



K= 11 has the best values of coherence, exclusivity and held-Out likelihood.

4.2.3 Run the analysis selecting k = 11

```
k = 11

mySTM <- stm(DfmStm$documents, vocab = DfmStm$vocab,
             K = k,
             prevalence = ~ party_id + populism + s(quarter),
             data = DfmStm$meta,
             init.type = "Spectral",
             verbose = TRUE)

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

4.2.4 Label topics

The frequency/exclusivity (FREX) scoring summarizes words according to their probability of appearance under a topic and the exclusivity to that topic. These words provide more semantically intuitive representations of each topic

```
labelTopics(mySTM, n=10)
```

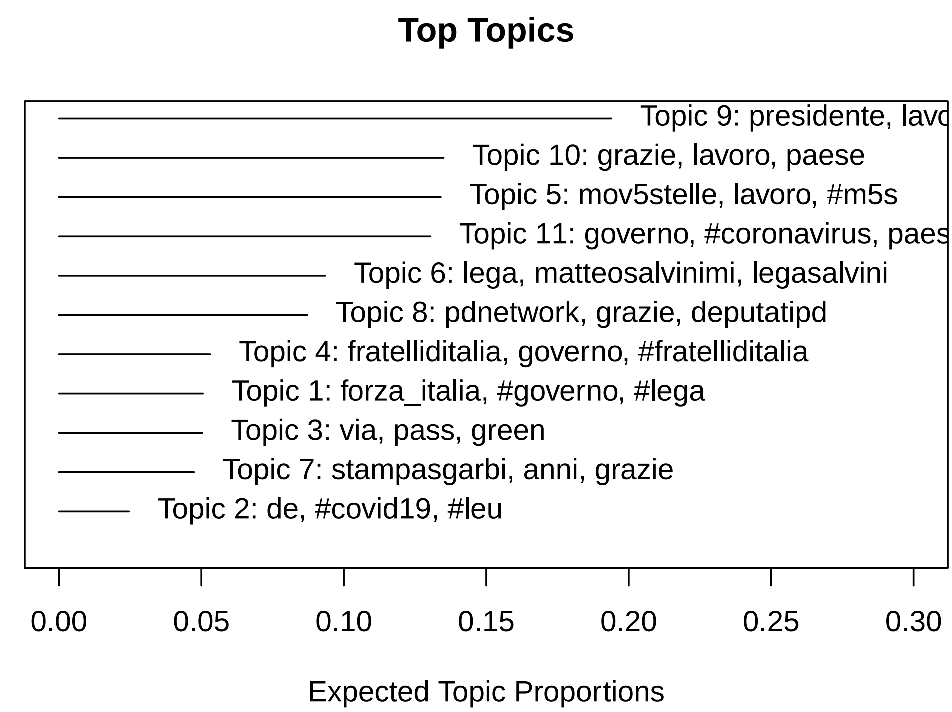
```
## Topic 1 Top Words:
##       Highest Prob: forza_italia, #governo, #lega, gruppoficamera, via, grazie, p
##       FREX: gruppoficamera, patriziarametta, pittoni, gruppofisenato, votalega, #
##       Lift: #grimoldi, normagi2, votalega, #abcdidanielasbrollini, #alternativo,
##       Score: forza_italia, gruppoficamera, pittoni, patriziarametta, #legasardegn
## Topic 2 Top Words:
##       Highest Prob: de, #covid19, #leu, italymfa, en, l'aggiornamento, bollettino
##       FREX: rapite, dall'oglio, #padredalloglio, paoladelusa, cooperazione_it, #s
##       Lift: #2030isnow, #africaeurope, #bamako, #ciudadania, #cncs, #corsaacasa,
##       Score: #padredalloglio, rapite, paoladelusa, cooperazione_it, dall'oglio, #
## Topic 3 Top Words:
##       Highest Prob: via, pass, green, draghi, rai, governo, fattoquotidiano, anni
##       FREX: gfi65, adginforma, massionline, angelazoppo, dottorbarbieri, andreagi
##       Lift: #datigrezzi, #laneuro, #manif28luglio, #prelemi, #rey, alternativa_it
##       Score: anzaldi, gfi65, massionline, adginforma, angelazoppo, ugambini, gian
## Topic 4 Top Words:
##       Highest Prob: fratelliditalia, governo, #fratelliditalia, italiani, via, gr
##       FREX: #fratelliditalia, vocedelpatriota, #giorgiameloni, #fdi, fratellidita
##       Lift: #orgogliotricolore, #patrioti, -staff, delmastro, #accalarentia, #ais
##       Score: fratelliditalia, vocedelpatriota, #fratelliditalia, fdi_parlamento,
## Topic 5 Top Words:
##       Highest Prob: mov5stelle, lavoro, #m5s, paese, legge, diretta, cittadini, c
```

```
##       FREX: m5s_senato, mov5stelle, #superbonus, #transizioneecologica, a_lisaco
##       Lift: #acqua2050, #annamariaparente, #anpr, #anticorruzione, #beigua, #bie
##       Score: mov5stelle, m5s_senato, #m5s, a_lisacorrado, greenitalia1, puglia_m
## Topic 6 Top Words:
##       Highest Prob: lega, matteosalvinimi, legasalvini, #lega, #salvini, salvini
##       FREX: lega_senato, angelociocca, #processateancheme, #oggivotolega, #prima
##       Lift: #accademiafederalelega, #aiutiamociacasanostra, #alballottaggiovotol
##       Score: #iostoconsalvini, legacamera, lega_senato, legasalvini, matteosalvi
## Topic 7 Top Words:
##       Highest Prob: stampasgarbi, anni, grazie, città, regione, grande, #venezia
##       FREX: stampasgarbi, comunevenezia, #governomusumeci, nino_ippolito, region
##       Lift: #venetodaamare, #abruzzesi, #antimafiafuffa, #barettasindaco, #buona
##       Score: stampasgarbi, #governomusumeci, nino_ippolito, comunevenezia, #orgo
## Topic 8 Top Words:
##       Highest Prob: pdnetwork, grazie, deputatipd, politica, bene, anni, legge,
##       FREX: azione_it, nomfup, piu_europa, graziano_delrio, elevisconti, adalucd
##       Lift: #cocaweb, #colapescedimartino, #dopofestival, #nanniespresso, #never
##       Score: deputatipd, azione_it, pdnetwork, senatoripd, nomfup, giusvapulejo,
## Topic 9 Top Words:
##       Highest Prob: presidente, lavoro, paese, grazie, grande, anni, donne, l'it
##       FREX: #ucraina, #patrickzaki, #romarinasce, #putin, legadilettanti, ucrain
##       Lift: #100m, #25novembre2021, #afirenzeperlapace, #agoràdemocratiche, #alt
##       Score: italiaviva, #romarinasce, pdnetwork, legadilettanti, #tokyo2020, #g
## Topic 10 Top Words:
##       Highest Prob: grazie, lavoro, paese, grande, cittadini, anni, #covid19, mo
##       FREX: gdf, #calabria, _carabinieri_, #palermo, esprimo, poliziadistato, #g
##       Lift: #massimotroisi, #matera2019, edraspa, tonello, #11giugno, #25giugno,
##       Score: #covid19, #gianlucarospi, #popoloprotagonista, _carabinieri_, #cala
## Topic 11 Top Words:
```

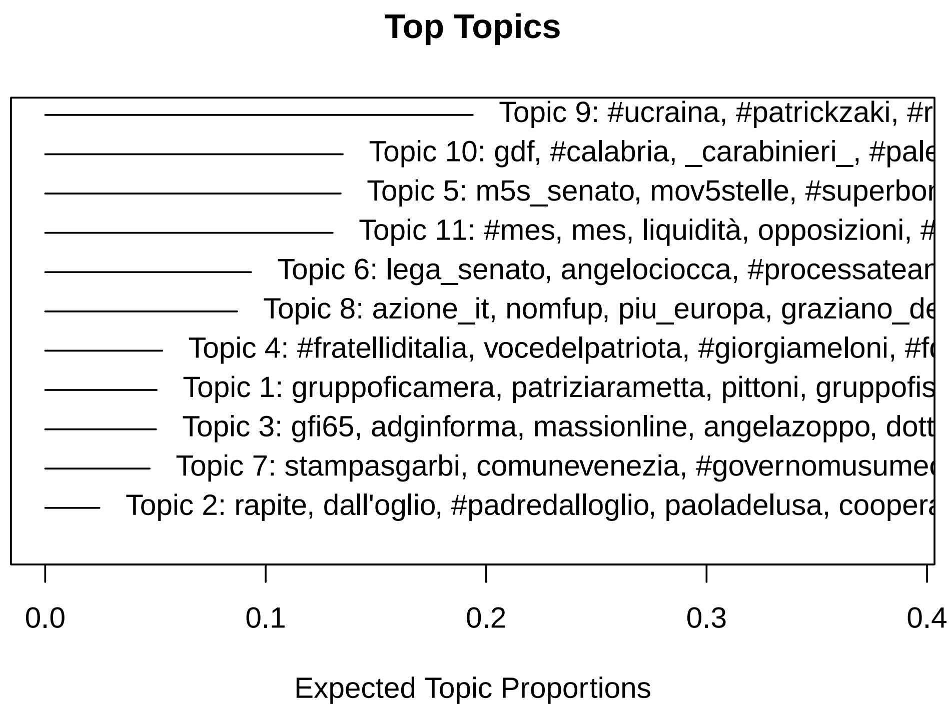
```
##       Highest Prob: governo, #coronavirus, paese, italiani, italia, l'italia, pol
##       FREX: #mes, mes, liquidità, opposizioni, #prescrizione, #italiaviva, #coro
##       Lift: #29febbraio, #annibali, #diamondprincess, #gennaroarma, #lamossadelc
##       Score: #coronavirus, italiaviva, #mes, mes, #fase2, liquidità, #covid2019,
```

4.2.5 Most frequent topic

```r
R <- plot(mySTM,
          type = "summary",
          xlim = c(0, .3))
```




```r
# plot just frex words for each topic
plot(mySTM, type = "summary", labeltype = c("frex"), n=5)  # topic 9 is the most fr
```



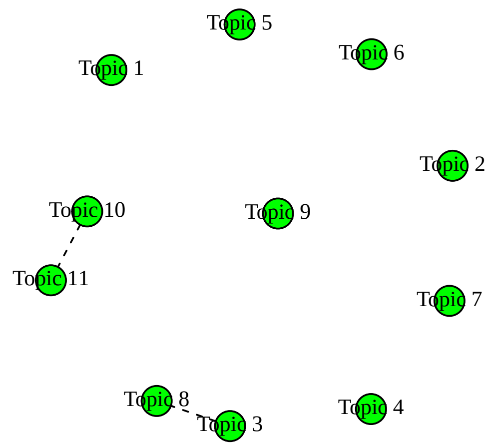
4.2.6 Find document most associated Text with the most frequent topic (9)

```
# Load original corpus
load("data/corpus.Rda")

# list the documents in the dfm
docs <- myDFM@Dimnames$docs

# Remove text with less than 1 word
corpus <- corpus_subset(corpus, ntoken(corpus) > 1)

# group the corpus like the dfm
```

```
corpus_g <- corpus_group(corpus,groups = interaction(nome, quarter,party_id))



# subset the same text of the dfm
subs_corpus <- corpus_subset(corpus_g, docnames(corpus_g) %in% docs)

documents <- as.character(subs_corpus)

documents <- as.vector(documents)

# Let's focus on topic 9
thought9 <- findThoughts(mySTM, texts=documents, topics=9, n=3)$docs[[1]]

#   qui non run esce errore seguente !!

# Error in findThoughts(mySTM, texts = texts(subs_corpus), topics = 9, n = 3) :
# Number of provided texts and number of documents modeled do not match
```

4.2.7 Correlation between topics

```
mod.out.corr <- topicCorr(mySTM)
plot(mod.out.corr)
```



4.2.8 Which are the the most likely topics across our documents?

```
#apply(mySTM$theta,1,which.max)


tab <- table(apply(mySTM$theta,1,which.max))
kable(tab[order(desc(tab))])
```

Var1	Freq
9	1309
5	809
11	707
10	652
6	631
8	463
4	309
1	245
3	245
7	207
2	133

4.2.9 save them back in the original dataframe

```
# STESSO PROBLEMA LIKE SOPRA:
# ORIGINAL CORPUS E STM NON SONO LO STESSO NUMERO
# 5710 vs 5713
subs_corpus$topic <- apply(mySTM$theta,1,which.max)

str(subs_corpus)

# Topic 5 - 5 random documents associated to it

set.seed(123)

sample(subs_corpus$text[subs_corpus$topic==5], 5)
```

4.3 Coefficients

```
#out$meta$rating <- as.factor(out$meta$rating)
prep <- estimateEffect(1:11 ~ party_id + populism + s(quarter),
                       mySTM,metadata = DfmStm$meta,
                       uncertainty = "Global")
summary(prep)
```

```
##
## Call:
## estimateEffect(formula = 1:11 ~ party_id + populism + s(quarter),
##     stmobj = mySTM, metadata = DfmStm$meta, uncertainty = "Global")
##
##
## Topic 1:
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             0.190725   0.018447  10.339  < 2e-16 ***
## party_idFDI            -0.140034   0.020417  -6.859 7.68e-12 ***
## party_idFI             -0.011609   0.018405  -0.631   0.5282
## party_idINDIPENDENTE -0.171896   0.028309  -6.072 1.34e-09 ***
## party_idIV             -0.124675   0.030036  -4.151 3.36e-05 ***
## party_idLEGA           -0.085065   0.018706  -4.547 5.54e-06 ***
## party_idLEU            -0.148444   0.022095  -6.719 2.01e-11 ***
## party_idM5S            -0.141948   0.018119  -7.834 5.59e-15 ***
## party_idMISTO          -0.111554   0.018983  -5.877 4.43e-09 ***
## party_idPD             -0.154919   0.017724  -8.741  < 2e-16 ***
## party_idREG_LEAGUES  -0.152064   0.028152  -5.402 6.88e-08 ***
```

```
## populism            -0.008643   0.001523  -5.677 1.44e-08 ***
## s(quarter)1          0.051605   0.132457   0.390   0.6968
## s(quarter)2          0.004390   0.058056   0.076   0.9397
## s(quarter)3         -0.003873   0.030220  -0.128   0.8980
## s(quarter)4         -0.017871   0.019907  -0.898   0.3694
## s(quarter)5         -0.029209   0.014478  -2.017   0.0437 *
## s(quarter)6         -0.021903   0.019411  -1.128   0.2592
## s(quarter)7         -0.018924   0.029885  -0.633   0.5266
## s(quarter)8         -0.036529   0.034163  -1.069   0.2850
## s(quarter)9         -0.018668   0.010689  -1.747   0.0808 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 2:
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.0240963  0.0137979   1.746 0.080800 .
## party_idFDI         -0.0038512  0.0146415  -0.263 0.792536
## party_idFI          -0.0028595  0.0136101  -0.210 0.833599
## party_idINDIPENDENTE  0.1336166  0.0299178   4.466 8.12e-06 ***
## party_idIV           0.0039485  0.0247786   0.159 0.873399
## party_idLEGA        -0.0054179  0.0132503  -0.409 0.682639
## party_idLEU         -0.0072656  0.0178174  -0.408 0.683449
## party_idM5S          0.0154623  0.0129847   1.191 0.233778
## party_idMISTO        0.0533434  0.0147772   3.610 0.000309 ***
## party_idPD           0.0226076  0.0131485   1.719 0.085597 .
## party_idREG_LEAGUES   0.1451007  0.0355056   4.087 4.44e-05 ***
```

```
## populism            -0.0068598  0.0014474  -4.739 2.20e-06 ***
## s(quarter)1          0.0474909  0.1095867   0.433 0.664767
## s(quarter)2         -0.0254836  0.0479738  -0.531 0.595302
## s(quarter)3          0.0096613  0.0265692   0.364 0.716149
## s(quarter)4         -0.0010323  0.0181295  -0.057 0.954596
## s(quarter)5         -0.0008385  0.0120987  -0.069 0.944748
## s(quarter)6          0.0015696  0.0183169   0.086 0.931716
## s(quarter)7         -0.0065401  0.0279764  -0.234 0.815170
## s(quarter)8          0.0077881  0.0335693   0.232 0.816546
## s(quarter)9          0.0063163  0.0102118   0.619 0.536254
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 3:
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.0092435  0.0156795   0.590   0.5555
## party_idFDI         -0.0060580  0.0171258  -0.354   0.7236
## party_idFI           0.0102323  0.0155696   0.657   0.5111
## party_idINDIPENDENTE -0.0498332  0.0276797  -1.800   0.0719 .
## party_idIV          -0.0213460  0.0271226  -0.787   0.4313
## party_idLEGA         0.0205457  0.0160654   1.279   0.2010
## party_idLEU         -0.0231779  0.0203157  -1.141   0.2540
## party_idM5S          0.0132186  0.0149348   0.885   0.3761
## party_idMISTO        0.0754295  0.0157202   4.798 1.64e-06 ***
## party_idPD          -0.0101944  0.0152452  -0.669   0.5037
## party_idREG_LEAGUES  -0.0328108  0.0265988  -1.234   0.2174
```

```
## populism              0.0009841  0.0017731   0.555   0.5789
## s(quarter)1          -0.0203254  0.1291961  -0.157   0.8750
## s(quarter)2           0.0149011  0.0569990   0.261   0.7938
## s(quarter)3          -0.0105891  0.0311940  -0.339   0.7343
## s(quarter)4           0.0303174  0.0206005   1.472   0.1412
## s(quarter)5           0.0119613  0.0141149   0.847   0.3968
## s(quarter)6           0.1445508  0.0251046   5.758 8.96e-09 ***
## s(quarter)7           0.0446011  0.0358703   1.243   0.2138
## s(quarter)8           0.1047823  0.0431660   2.427   0.0152 *
## s(quarter)9           0.0625024  0.0118577   5.271 1.41e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 4:
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -0.0178879  0.0111110  -1.610 0.107471
## party_idFDI           0.4513405  0.0174825  25.817  < 2e-16 ***
## party_idFI            0.0454711  0.0113961   3.990 6.69e-05 ***
## party_idINDIPENDENTE -0.0062589  0.0208235  -0.301 0.763755
## party_idIV           -0.0063568  0.0193566  -0.328 0.742619
## party_idLEGA          0.0170733  0.0110033   1.552 0.120801
## party_idLEU           0.0169227  0.0152916   1.107 0.268485
## party_idM5S           0.0107626  0.0104781   1.027 0.304394
## party_idMISTO         0.0415945  0.0117438   3.542 0.000401 ***
## party_idPD            0.0029460  0.0105926   0.278 0.780934
## party_idREG_LEAGUES   0.0098415  0.0207680   0.474 0.635605
```

```
## populism             0.0156521  0.0021551   7.263 4.31e-13 ***
## s(quarter)1          0.0323322  0.0995359   0.325 0.745322
## s(quarter)2         -0.0031862  0.0438591  -0.073 0.942090
## s(quarter)3          0.0321969  0.0248566   1.295 0.195268
## s(quarter)4          0.0641008  0.0165334   3.877 0.000107 ***
## s(quarter)5          0.0257806  0.0108376   2.379 0.017401 *
## s(quarter)6         -0.0073199  0.0161843  -0.452 0.651083
## s(quarter)7          0.0659624  0.0235743   2.798 0.005158 **
## s(quarter)8         -0.0633661  0.0277408  -2.284 0.022395 *
## s(quarter)9          0.0005911  0.0081236   0.073 0.942001
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 5:
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.034194   0.018880   1.811 0.070186 .
## party_idFDI         -0.003332   0.019455  -0.171 0.864036
## party_idFI           0.023307   0.017842   1.306 0.191498
## party_idINDIPENDENTE  0.215220   0.042256   5.093 3.63e-07 ***
## party_idIV           0.098809   0.034980   2.825 0.004748 **
## party_idLEGA         0.003717   0.017715   0.210 0.833800
## party_idLEU          0.137799   0.026714   5.158 2.58e-07 ***
## party_idM5S          0.235287   0.017411  13.514  < 2e-16 ***
## party_idMISTO        0.071424   0.018370   3.888 0.000102 ***
## party_idPD           0.047387   0.017280   2.742 0.006119 **
## party_idREG_LEAGUES   0.042825   0.037211   1.151 0.249825
```

```
## populism             -0.007342   0.002234  -3.286 0.001022 **
## s(quarter)1          -0.087316   0.158631  -0.550 0.582044
## s(quarter)2           0.013665   0.070109   0.195 0.845473
## s(quarter)3           0.060733   0.037687   1.611 0.107130
## s(quarter)4          -0.055346   0.023022  -2.404 0.016246 *
## s(quarter)5           0.053370   0.020477   2.606 0.009177 **
## s(quarter)6          -0.058525   0.026149  -2.238 0.025252 *
## s(quarter)7           0.209564   0.042837   4.892 1.02e-06 ***
## s(quarter)8          -0.192113   0.047584  -4.037 5.48e-05 ***
## s(quarter)9           0.007735   0.014751   0.524 0.600056
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 6:
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.042102   0.015707   2.681 0.007372 **
## party_idFDI          -0.016551   0.016486  -1.004 0.315470
## party_idFI           -0.011165   0.015085  -0.740 0.459231
## party_idINDIPENDENTE -0.025420   0.028086  -0.905 0.365473
## party_idIV           -0.021646   0.025832  -0.838 0.402075
## party_idLEGA          0.385872   0.016192  23.830  < 2e-16 ***
## party_idLEU          -0.016974   0.019687  -0.862 0.388606
## party_idM5S           0.001058   0.014302   0.074 0.941015
## party_idMISTO        -0.012542   0.015373  -0.816 0.414614
## party_idPD           -0.015042   0.014533  -1.035 0.300700
## party_idREG_LEAGUES  -0.001621   0.027411  -0.059 0.952857
```

```
## populism              0.012548   0.002292   5.475 4.57e-08 ***
## s(quarter)1          -0.082477   0.142831  -0.577 0.563663
## s(quarter)2          -0.010643   0.061357  -0.173 0.862300
## s(quarter)3           0.040077   0.035015   1.145 0.252442
## s(quarter)4          -0.075153   0.020712  -3.628 0.000288 ***
## s(quarter)5           0.030592   0.017330   1.765 0.077567 .
## s(quarter)6          -0.034046   0.022979  -1.482 0.138488
## s(quarter)7          -0.052179   0.035092  -1.487 0.137093
## s(quarter)8          -0.058081   0.037991  -1.529 0.126370
## s(quarter)9          -0.040076   0.011951  -3.353 0.000804 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 7:
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1.569e-01  1.996e-02   7.860 4.56e-15 ***
## party_idFDI           -7.637e-02  2.233e-02  -3.420 0.000631 ***
## party_idFI            -8.308e-02  2.070e-02  -4.013 6.08e-05 ***
## party_idINDIPENDENTE  1.114e-01  3.687e-02   3.021 0.002528 **
## party_idIV            -1.345e-01  2.925e-02  -4.598 4.36e-06 ***
## party_idLEGA          -7.708e-02  2.011e-02  -3.832 0.000128 ***
## party_idLEU           -1.193e-01  2.445e-02  -4.879 1.10e-06 ***
## party_idM5S           -1.209e-01  1.971e-02  -6.135 9.11e-10 ***
## party_idMISTO         -1.070e-01  2.026e-02  -5.279 1.35e-07 ***
## party_idPD            -9.431e-02  1.975e-02  -4.775 1.84e-06 ***
## party_idREG_LEAGUES   1.899e-02  3.968e-02   0.479 0.632293
```

```
## populism             -8.086e-03  1.619e-03  -4.993 6.12e-07 ***
## s(quarter)1          -2.218e-01  1.252e-01  -1.771 0.076594 .
## s(quarter)2           5.329e-02  5.504e-02   0.968 0.333044
## s(quarter)3           1.650e-03  3.030e-02   0.054 0.956576
## s(quarter)4          -3.088e-02  1.969e-02  -1.569 0.116801
## s(quarter)5          -1.975e-05  1.590e-02  -0.001 0.999009
## s(quarter)6           6.910e-02  2.193e-02   3.151 0.001636 **
## s(quarter)7          -6.219e-02  3.236e-02  -1.922 0.054712 .
## s(quarter)8           1.404e-02  3.633e-02   0.387 0.699073
## s(quarter)9           2.731e-02  1.177e-02   2.320 0.020360 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 8:
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.084574   0.019215   4.401  1.1e-05 ***
## party_idFDI          -0.005658   0.020531  -0.276 0.782861
## party_idFI           -0.014923   0.019153  -0.779 0.435936
## party_idINDIPENDENTE -0.098924   0.033176  -2.982 0.002878 **
## party_idIV           -0.012537   0.034729  -0.361 0.718115
## party_idLEGA         -0.059670   0.019214  -3.106 0.001908 **
## party_idLEU           0.057184   0.027071   2.112 0.034697 *
## party_idM5S          -0.050148   0.018381  -2.728 0.006385 **
## party_idMISTO         0.022403   0.020011   1.120 0.262966
## party_idPD            0.067000   0.019400   3.454 0.000557 ***
## party_idREG_LEAGUES  -0.034375   0.034465  -0.997 0.318624
```

```
## populism             0.001519   0.001951   0.779 0.436061
## s(quarter)1         -0.373088   0.152163  -2.452 0.014241 *
## s(quarter)2          0.169888   0.066832   2.542 0.011047 *
## s(quarter)3         -0.072317   0.035999  -2.009 0.044602 *
## s(quarter)4          0.063879   0.024315   2.627 0.008633 **
## s(quarter)5          0.007634   0.017982   0.425 0.671195
## s(quarter)6          0.008651   0.025218   0.343 0.731566
## s(quarter)7          0.029230   0.040972   0.713 0.475621
## s(quarter)8          0.016353   0.046882   0.349 0.727246
## s(quarter)9          0.003070   0.013391   0.229 0.818663
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 9:
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.079964   0.022235   3.596 0.000325 ***
## party_idFDI          -0.120279   0.023495  -5.119 3.17e-07 ***
## party_idFI           -0.022564   0.022083  -1.022 0.306925
## party_idINDIPENDENTE -0.040564   0.048447  -0.837 0.402465
## party_idIV            0.140044   0.044230   3.166 0.001552 **
## party_idLEGA         -0.112169   0.021543  -5.207 1.99e-07 ***
## party_idLEU           0.074285   0.030973   2.398 0.016499 *
## party_idM5S          -0.024584   0.021314  -1.153 0.248786
## party_idMISTO        -0.039088   0.023802  -1.642 0.100595
## party_idPD            0.133881   0.023085   5.800 7.01e-09 ***
## party_idREG_LEAGUES  -0.028586   0.037887  -0.755 0.450564
```

```
## populism              -0.000705   0.002260  -0.312 0.755143
## s(quarter)1           -1.118906   0.164407  -6.806 1.11e-11 ***
## s(quarter)2            0.435717   0.072583   6.003 2.06e-09 ***
## s(quarter)3           -0.166432   0.040374  -4.122 3.81e-05 ***
## s(quarter)4            0.175209   0.028404   6.168 7.37e-10 ***
## s(quarter)5            0.030861   0.019118   1.614 0.106534
## s(quarter)6            0.335293   0.030852  10.868  < 2e-16 ***
## s(quarter)7           -0.081241   0.052244  -1.555 0.119995
## s(quarter)8            0.647119   0.057623  11.230  < 2e-16 ***
## s(quarter)9            0.202032   0.015833  12.760  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 10:
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)            0.135183   0.018013   7.505 7.10e-14 ***
## party_idFDI           -0.042129   0.019169  -2.198 0.028001 *
## party_idFI             0.020448   0.018402   1.111 0.266523
## party_idINDIPENDENTE  0.003060   0.034477   0.089 0.929286
## party_idIV             0.014743   0.030275   0.487 0.626307
## party_idLEGA          -0.059454   0.017469  -3.403 0.000670 ***
## party_idLEU            0.016137   0.023050   0.700 0.483902
## party_idM5S            0.075745   0.017816   4.251 2.16e-05 ***
## party_idMISTO         -0.022640   0.019159  -1.182 0.237379
## party_idPD            -0.007908   0.017773  -0.445 0.656402
## party_idREG_LEAGUES   0.051312   0.036479   1.407 0.159603
```

```
## populism             -0.004059   0.001781  -2.279 0.022683 *
## s(quarter)1           1.350315   0.145088   9.307  < 2e-16 ***
## s(quarter)2          -0.507570   0.063516  -7.991 1.61e-15 ***
## s(quarter)3           0.197423   0.036375   5.427 5.95e-08 ***
## s(quarter)4          -0.032380   0.024499  -1.322 0.186329
## s(quarter)5           0.100391   0.018161   5.528 3.39e-08 ***
## s(quarter)6          -0.208555   0.022001  -9.479  < 2e-16 ***
## s(quarter)7           0.088432   0.035241   2.509 0.012122 *
## s(quarter)8          -0.230963   0.039167  -5.897 3.92e-09 ***
## s(quarter)9          -0.043505   0.011727  -3.710 0.000209 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Topic 11:
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           0.261138   0.017303  15.092  < 2e-16 ***
## party_idFDI          -0.037329   0.017620  -2.119  0.03416 *
## party_idFI            0.046592   0.016624   2.803  0.00508 **
## party_idINDIPENDENTE -0.071182   0.029682  -2.398  0.01651 *
## party_idIV            0.063320   0.031634   2.002  0.04537 *
## party_idLEGA         -0.028843   0.015837  -1.821  0.06863 .
## party_idLEU           0.012454   0.020762   0.600  0.54866
## party_idM5S          -0.014246   0.015390  -0.926  0.35463
## party_idMISTO         0.028246   0.016496   1.712  0.08690 .
## party_idPD            0.008302   0.015786   0.526  0.59896
## party_idREG_LEAGUES  -0.018421   0.031182  -0.591  0.55471
```

```
## populism              0.005043   0.001929   2.615  0.00895 **
## s(quarter)1           0.419686   0.140771   2.981  0.00288 **
## s(quarter)2          -0.143947   0.060938  -2.362  0.01820 *
## s(quarter)3          -0.089103   0.033416  -2.666  0.00769 **
## s(quarter)4          -0.120523   0.021082  -5.717 1.14e-08 ***
## s(quarter)5          -0.230321   0.014344 -16.057  < 2e-16 ***
## s(quarter)6          -0.228700   0.020292 -11.270  < 2e-16 ***
## s(quarter)7          -0.216925   0.030126  -7.201 6.77e-13 ***
## s(quarter)8          -0.207864   0.034871  -5.961 2.66e-09 ***
## s(quarter)9          -0.207389   0.012606 -16.451  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.4 Interpretation and validation

Da sistemare

```
#Topic Prevalence Over-time
#qui potresti far vedere che la prevalence di alcuni topics ha dei picchi
#in periodo in cui ha senso che abbia dei picchi...
#Anche questo vale come validation, va runnato però dopo che hai runnato estimateE
#perché ti servono i coefficienti dei quarters della regression...
#Anche qui, esempio con Topic 1 poi devi guardare tutti i topics...
#Va un po' sistemato (vedi x axis), magari da provare con GGPLOT...
plot(prep, "quarter", method = "continuous", topics = 1,
       model = z, printlegend = FALSE, xaxt = "n", xlab = "Time")
quartseq <- seq(from = as.Date("2020-01-01"),
                  to = as.Date("2022-04-18"), by = "quarter")
quartnames <- quarters(quartseq)
```

```r
axis(1,at = as.numeric(quartseq) - min(as.numeric(quartseq)),
     labels = quartnames)
```

5 FER: Facial Emotion Recognition Analysis

5.1 Report on the analysis made with FER Python package

As an additional analysis, we have developed an application that implements an emotion recognition system in videos. The aim is to investigate which were the main emotions conveyed by Italian party leaders through interviews and live broadcasts published on their respective facebook profiles, during the first two months after the invasion of Ukraine by the Russian army.

The corpus for this analysis was constructed by manually searching for videos showing the politician engaged in a live broadcast or interview on the pages of the following party leaders: Forza Italia: Silvio Berlusconi; Fratelli d'Italia: Giorgia Meloni; Italia Viva: Matteo Renzi; Lega: Matteo Salvini; Liberi e Uguali: Roberto Speranza; Movimento 5 stelle: Giuseppe Conte; Partito Democratico: Enrico Letta;

Subsequently, only videos that were suitable for analysis were downloaded, i.e. videos with a frontal shot of the politician and not exceeding 35 MB in size; for some party leaders, no video suitable for analysis was found. A total of 21 videos were collected, distributed in this way: Silvio Berlusconi: 0 Giorgia Meloni: 7 Matteo Renzi: 2 Matteo Salvini: 3 Roberto Speranza: 0 Giuseppe Conte: 7 Enrico Letta: 2

To perform this analysis we used the Python FER (Face Emotion Recognition) package, developed by Justin Shenk using the FER2013 dataset curated 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 package allows you to call a keras convolutional neural network model trained using the dataset FER2013, described in "Challenges in Representation Learning: A report on three machine learning contests". In order to simplify the use of the package and allow access to it everywhere (not exclusively on PCs with python installed), it was decided to develop a simple cloud-hosted application. For this solution, I used the freemwork streamlit, which offers free application hosting to the developer community. Thanks to this solution, I was able to speed up analysis times and expand the user base of the FER package to include users who do not use the Python language. Through the application it is possible to upload a video in mp4 format (max 35 MB) and call the model to perform the analysis. Once the analysis is performed, a summary graph of the emotions detected frame by frame and a table showing the proportion of each emotion detected with respect to the length of the video are displayed. Finally, a button is available with which to download the results in a .csv file in which the coordinates of the faces detected and the proportion detected frame by frame for each emotion are shown, the file can be saved on any device and re-analysed with any software that allows the processing of .csv files. The code is free and available on the project's github repository.

5.2 Import the datasets

```
# CONTE
Conte_07_03_22_00 <- read_csv("data/video_emotions/Conte_07-03-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```
Conte_09_03_22_00 <- read_csv("data/video_emotions/Conte_09-03-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```
Conte_22_02_22_00 <- read_csv("data/video_emotions/Conte_22-02-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Conte_23_02_22_00 <- read_csv("data/video_emotions/Conte_23-02-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Conte_23_02_22_01 <- read_csv("data/video_emotions/Conte_23-02-22_01.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Conte_24_02_22_01 <- read_csv("data/video_emotions/Conte_24-02-22_01.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Conte_28_02_22_00 <- read_csv("data/video_emotions/Conte_28-02-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
# LETTA
Letta_03_03_22_00 <- read_csv("data/video_emotions/Letta_03-03-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Letta_06_04_22_00 <- read_csv("data/video_emotions/Letta_06-04-22_00.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```
# MELONI
Meloni_1_03_2022 <- read_csv("data/video_emotions/Meloni_1-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```
Meloni_11_03_2022_02 <- read_csv("data/video_emotions/Meloni_11-03-2022_02.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```
Meloni_11_03_2022 <- read_csv("data/video_emotions/Meloni_11-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Meloni_15_03_2022 <- read_csv("data/video_emotions/Meloni_15-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Meloni_22_03_2022 <- read_csv("data/video_emotions/Meloni_22-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Meloni_29_03_2022 <- read_csv("data/video_emotions/Meloni_29-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Meloni_31_03_2022<- read_csv("data/video_emotions/Meloni_31-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
# RENZI
Renzi_19_04_2022 <- read_csv("data/video_emotions/Renzi_19-04-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Renzi_30_03_2022 <- read_csv("data/video_emotions/Renzi_30-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
# SALVINI
Salvini_08_03_2022 <- read_csv("data/video_emotions/Salvini_08-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Salvini_08_04_2022_02 <- read_csv("data/video_emotions/Salvini_08-04-2022_02.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

```r
Salvini_16_03_2022 <- read_csv("data/video_emotions/Salvini_16-03-2022.csv",
    col_types = cols(angry = col_number(),
        disgust = col_number(), fear = col_number(),
        happy = col_number(), sad = col_number(),
        surprise = col_number(), neutral = col_number()))
```

```
## New names:
## * `` -> `...1`
```

5.3 Analyse Conte datasets

```
#1
# Conte_07_03_22_00
Conte_07_03_22_00_prop <- c(
  angry <-  sum(Conte_07_03_22_00$angry),
  disgust <- sum(Conte_07_03_22_00$disgust),
  fear <- sum(Conte_07_03_22_00$fear),
  happy <- sum(Conte_07_03_22_00$happy),
  sad <- sum(Conte_07_03_22_00$sad),
  surprise <- sum(Conte_07_03_22_00$surprise),
  meutral <- sum(Conte_07_03_22_00$neutral)
)
```

```
#2
# Conte_09_03_22_00
Conte_09_03_22_00_prop <- c(
  angry <-  sum(Conte_09_03_22_00$angry),
  disgust <- sum(Conte_09_03_22_00$disgust),
  fear <- sum(Conte_09_03_22_00$fear),
  happy <- sum(Conte_09_03_22_00$happy),
  sad <- sum(Conte_09_03_22_00$sad),
  surprise <- sum(Conte_09_03_22_00$surprise),
  meutral <- sum(Conte_09_03_22_00$neutral)
)
```

```
#3
# Conte_22_02_22_00
i = Conte_22_02_22_00
```

```
Conte_22_02_22_00_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#4
# Conte_23_02_22_00
i = Conte_23_02_22_00
Conte_23_02_22_00_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#5
# Conte_23_02_22_01
i = Conte_23_02_22_01
Conte_23_02_22_01_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
```

```
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#6
# Conte_24_02_22_01
i = Conte_24_02_22_01
Conte_24_02_22_01_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#7
# Conte_28_02_22_00
i = Conte_28_02_22_00
Conte_28_02_22_00_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
```

```
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
conte <- rbind(Conte_07_03_22_00_prop,
               Conte_09_03_22_00_prop,
               Conte_22_02_22_00_prop,
               Conte_23_02_22_00_prop,
               Conte_23_02_22_01_prop,
               Conte_24_02_22_01_prop,
               Conte_28_02_22_00_prop
               )
emo_label <- colnames(Conte_07_03_22_00)[3:9]

colnames(conte) <- emo_label

conte <- as.data.frame(conte)

tot_conte <- max(Conte_07_03_22_00$...1) +
              max(Conte_09_03_22_00$...1) +
              max(Conte_22_02_22_00$...1) +
              max(Conte_23_02_22_00$...1) +
              max(Conte_23_02_22_01$...1) +
              max(Conte_24_02_22_01$...1) +
              max(Conte_28_02_22_00$...1)


conte[8,] <- c(sum(conte$angry)/tot_conte * 100, sum(conte$disgust)/tot_conte *100,
               sum(conte$fear)/tot_conte *100, sum(conte$happy)/tot_conte *100,
               sum(conte$sad)/tot_conte *100 ,sum(conte$surprise)/tot_conte * 100,
```

```
                sum(conte$neutral)/tot_conte *100)
```

5.4 Analyse Letta datasets

```
#1
# Letta_03_03_22_00
i = Letta_03_03_22_00
Letta_03_03_22_00_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#2
# Letta_06_04_22_00
i = Letta_06_04_22_00
Letta_06_04_22_00_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
```

```
)
```

```
letta <- rbind(Letta_03_03_22_00_prop,
               Letta_06_04_22_00_prop
               )
colnames(letta) <- emo_label

letta <- as.data.frame(letta)

tot_letta <- max(Letta_03_03_22_00$...1) +
              max(Letta_06_04_22_00$...1)

letta[3,] <- c(sum(letta$angry)/tot_letta * 100, sum(letta$disgust)/tot_letta *100,
               sum(letta$fear)/tot_letta *100, sum(letta$happy)/tot_letta *100,
               sum(letta$sad)/tot_letta *100 ,sum(letta$surprise)/tot_letta * 100,
               sum(letta$neutral)/tot_letta *100)
```

5.5 Analyse Meloni datasets

```
#1
# Meloni_1_03_2022
i = Meloni_1_03_2022
Meloni_1_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
```

```
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#2
# Meloni_11_03_2022_02
i = Meloni_11_03_2022_02
Meloni_11_03_2022_02_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#3
# Meloni_11_03_2022
i = Meloni_11_03_2022
Meloni_11_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#4
# Meloni_15_03_2022
i = Meloni_15_03_2022
Meloni_15_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#5
# Meloni_22_03_2022
i = Meloni_22_03_2022
Meloni_22_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#6
# Meloni_29_03_2022
i = Meloni_29_03_2022
```

```r
Meloni_29_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```r
#7
# Meloni_31_03_2022
i = Meloni_31_03_2022
Meloni_31_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```r
meloni <- rbind(Meloni_1_03_2022_prop,
               Meloni_11_03_2022_02_prop,
               Meloni_11_03_2022_prop,
               Meloni_15_03_2022_prop,
               Meloni_22_03_2022_prop,
               Meloni_29_03_2022_prop,
```

```
               Meloni_31_03_2022_prop
               )
colnames(meloni) <- emo_label

meloni <- as.data.frame(meloni)



tot_meloni <- max(Meloni_1_03_2022$...1) +
              max(Meloni_11_03_2022_02$...1)+
              max(Meloni_11_03_2022$...1)+
              max(Meloni_15_03_2022$...1)+
              max(Meloni_22_03_2022$...1)+
              max(Meloni_29_03_2022$...1)+
              max(Meloni_31_03_2022$...1)

meloni[8,] <- c(sum(meloni$angry)/tot_meloni * 100, sum(meloni$disgust)/tot_meloni *
               sum(meloni$fear)/tot_meloni *100, sum(meloni$happy)/tot_meloni *100,
               sum(meloni$sad)/tot_meloni *100 ,sum(meloni$surprise)/tot_meloni * 10
               sum(meloni$neutral)/tot_meloni *100)
```

5.6 Analyse Renzi datasets

```
#1
# Renzi_19_04_2022
i = Renzi_19_04_2022
Renzi_19_04_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
```

```
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#2
# Renzi_30_03_2022
i = Renzi_30_03_2022
Renzi_30_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
renzi <- rbind(Renzi_19_04_2022_prop,
               Renzi_30_03_2022_prop
               )
colnames(renzi) <- emo_label


renzi <- as.data.frame(renzi)


tot_renzi <- max(Renzi_19_04_2022$...1) +
              max(Renzi_30_03_2022$...1)
```

```
renzi[3,] <- c(sum(renzi$angry)/tot_renzi * 100, sum(renzi$disgust)/tot_renzi *100,
               sum(renzi$fear)/tot_renzi *100, sum(renzi$happy)/tot_renzi *100,
               sum(renzi$sad)/tot_renzi *100 ,sum(renzi$surprise)/tot_renzi * 100,
               sum(renzi$neutral)/tot_renzi *100)
```

5.7 Analyse Salvini datasets

```
#1
# Salvini_08_03_2022
i = Salvini_08_03_2022
Salvini_08_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#2
# Salvini_08_04_2022_02
i = Salvini_08_04_2022_02
Salvini_08_04_2022_02_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
```

```
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
#3
# Salvini_16_03_2022
i = Salvini_16_03_2022
Salvini_16_03_2022_prop <- c(
  angry <-  sum(i$angry),
  disgust <- sum(i$disgust),
  fear <- sum(i$fear),
  happy <- sum(i$happy),
  sad <- sum(i$sad),
  surprise <- sum(i$surprise),
  meutral <- sum(i$neutral)
)
```

```
salvini <- rbind(Salvini_08_03_2022_prop,
               Salvini_08_04_2022_02_prop,
               Salvini_16_03_2022_prop
               )
colnames(salvini) <- emo_label


salvini <- as.data.frame(salvini)


tot_salvini <- max(Salvini_08_03_2022$...1) +
                max(Salvini_08_04_2022_02$...1)+
```

```
              max(Salvini_16_03_2022$...1)

salvini[4,] <- c(sum(salvini$angry)/tot_salvini * 100, sum(salvini$disgust)/tot_salv
               sum(salvini$fear)/tot_salvini *100, sum(salvini$happy)/tot_salvini *1
               sum(salvini$sad)/tot_salvini *100 ,sum(salvini$surprise)/tot_salvini
               sum(salvini$neutral)/tot_salvini *100)
```

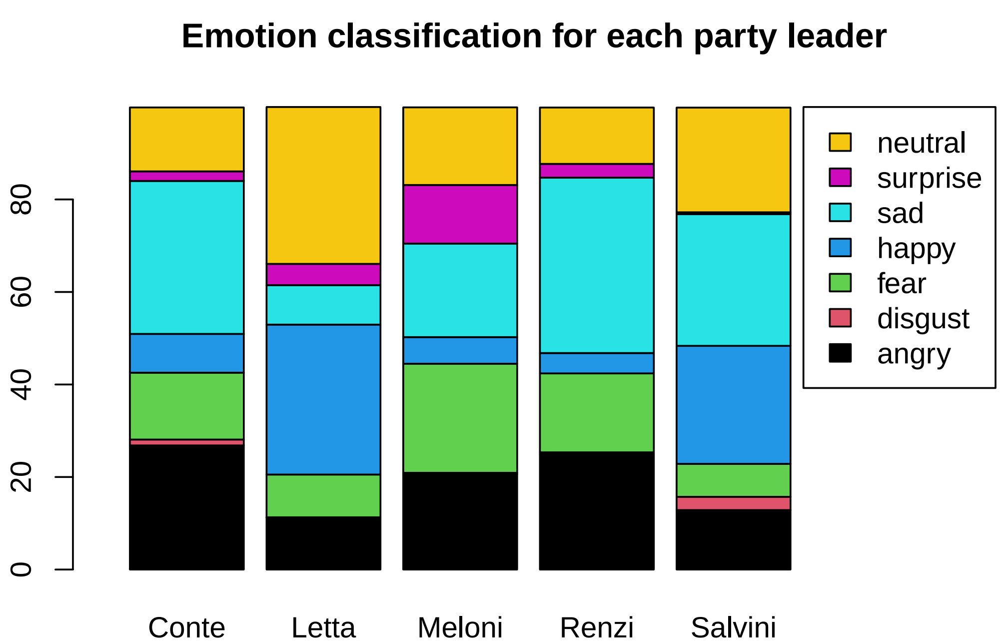
5.8 Create dataset with the proportion of the emotions registered for each leader

```
emotions <- rbind(conte[8,], letta[3,], meloni[8,],
                  renzi [3,], salvini[4,])

emotions <- as.data.frame(emotions)

rownames(emotions) <- c("Conte", "Letta","Meloni", "Renzi", "Salvini")

kable(emotions)
```

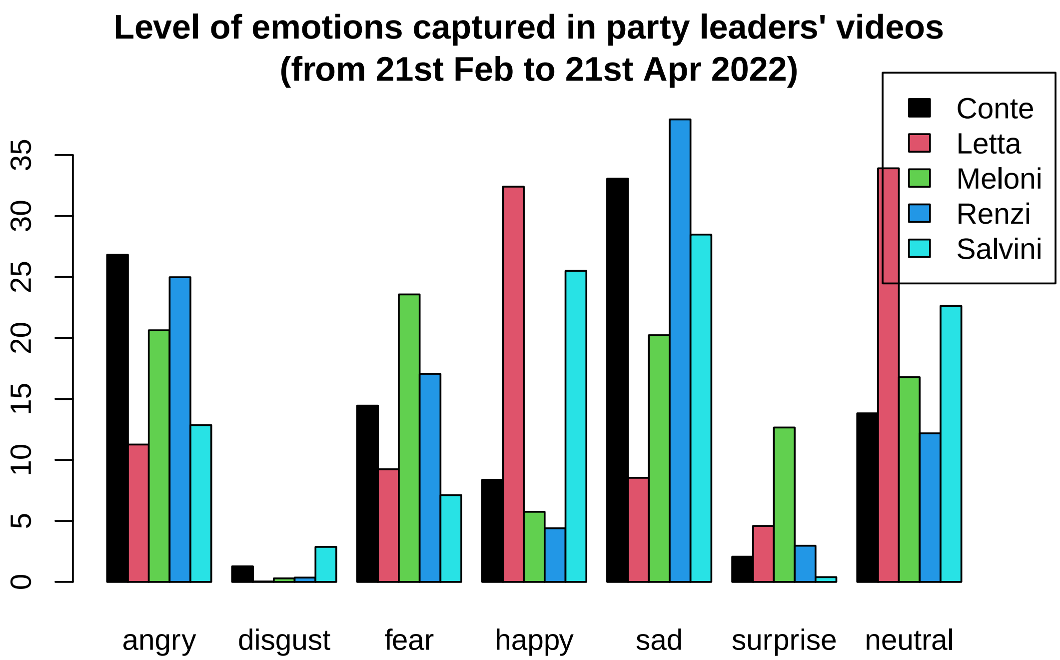
	angry	disgust	fear	happy	sad	surprise	neutral
Conte	26.82598	1.2745357	14.453157	8.376083	33.067107	2.0669418	13.82099
Letta	11.26310	0.0342991	9.236472	32.409459	8.532808	4.5901150	33.91223
Meloni	20.63115	0.2883622	23.570927	5.747407	20.226604	12.6588713	16.78353
Renzi	24.98139	0.3537467	17.060469	4.398206	37.923233	2.9666313	12.18218
Salvini	12.85566	2.8709759	7.116432	25.509843	28.479119	0.3918993	22.63165

5.9 Results

```r
matrix <- as.matrix(emotions)



barplot(t(matrix),
        col = 1:ncol(matrix),
        legend.text = TRUE,
        args.legend = list(x = "topright"),
         xlim=c(0,7.5),
        main = "Emotion classification for each party leader"
        )
```



```
barplot(matrix,
        col = 1:nrow(matrix),
        legend.text = TRUE,
        args.legend = list(x = "topright",
                           inset = c(- 0.065, -0.1)),
        beside = T,
        main = "Level of emotions captured in party leaders' videos \n (from 21st Fe
```



Looking at these results, it is possible to observe which emotions were used by party leaders during the first two months of the Ukrainian invasion, it should be noted that the videos were downloaded regardless of whether the topic at hand was the war in Ukraine. Looking at the data, it can be seen that the most recorded emotion is Sad for politicians Giuseppe Conte, Matteo Renzi and Matteo Salvini; while for Enrico Letta the highest percentage is Neutral and for Giorgia Meloni it is Fear. The major limitation of this analysis is the model's low reliability in extracting emotions correctly, the accuracy percentage was recorded at around 66% as indicated in the paper by Octavio Arriaga, Matias Valdenegro-Toro, Paul Plöger (Real-time Convolutional Neural Networks for Emotion and Gender Classification); and the second major limitation is the analysis time, each video takes about two and a half times the length of the video itself. To obtain more reliable results, as reported on other studies, it is possible to use Microsoft's Cognitive Services with which it is possible to replicate the same analysis by calling up an API, the disadvantage in this case being that it is a proprietary paid service based on a non-open source and non-accessible model. The advantage of using an open source model such as FER is given by the possibility, through future studies, of replicating the analysis using models trained on different datasets with a view to defining an increasingly accurate neural network model for a specific use such as this.