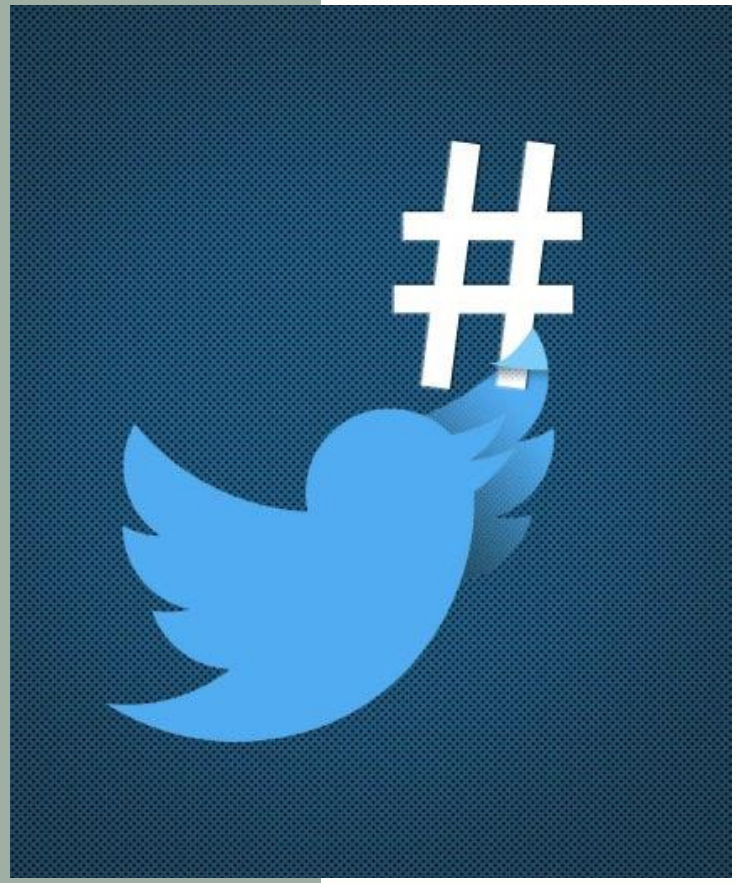


Green Pass: Social Interaction on Twitter

Andrea Munarin
Ca' Foscari University, Venice, Italy

Generalità del progetto

La creazione dei dataset è stata realizzata in Pyhton
La visualizzazione dei dati invece è stata realizzata in R

Le librerie utilizzate sono:

- Igraph e ggplot2 in R per costruire grafi e grafici
- TwitterAPI e pandas per la gestione dei dataframe in R
- Sys e re per alcune funzionalità python
- feel_it è una libreria per l'analisi del sentiment di frasi italiane sviluppata appositivamente per i tweet

Obiettivi del progetto

1. Studio di polarizzazione e presenza echo chambers relative al tema Green Pass, in Italia, nell'ultimo mese
2. Analisi dei tweet recenti forniti dall'API
3. Analisi delle timeline degli utenti
4. Utilizzo del sentiment per determinare le posizioni degli utenti in riferimento al tema

Bias

Importante considerare presenza di Bias dovuti a:

- Limite sul numero di tweet e sul periodo posti dall'API ufficiale di Twitter
- Utenti twitter non sono rappresentativi della popolazione italiana
- Bias proveniente da cleaning ed enrichment dei dataset
- Bias proveneninete dalla sorgente dei dati (Twitter)

Source

Sorgente selezionata: Twitter

- Facilità di ottenimento dei dati dell'ultimo periodo
- Possibile scaricare timeline degli utenti e determinare informazioni sui loro profili
- Gestione delle conversation: ogni tweet ha un id della conversazione grazie a questo è possibile determinare una catena di tweet correlati al primo

01
COLLEZIONARE I DATI

Python

Recent Tweets

```
tweet = api.request('tweets/search/recent',
                {'query':QUERY_POSITIVE,
                 'max_results': "100",
                 'tweet.fields': "public_metrics,conversation_id"}).json()

next_token = tweet['meta']['next_token']

for t in tweet['data']:
   t['public_metrics'] = int(t['public_metrics']['like_count'] + t['public_metrics']['quote_count'] +
t['public_metrics']['reply_count'] +t['public_metrics']['retweet_count'])
   t['text'] = str(sentiment_classifier.predict([t['text']]))
   if tweet_data.empty:
      tweet_data = pd.DataFrame(t, index=[0])
      init = False
   else:
      aux = pd.DataFrame(t, index=[0])
      tweet_data = tweet_data.append(aux, ignore_index=True)
```

Tweets selezionati

```
tweet_data.sort_values('public_metrics', ascending = False, ignore_index = True, inplace=True)

for i in tweet_data.index:
   # conta se positivo e negativo
   if (b_positive and positive <= 5 ) or (not positive  and negative <= 5):
      if selected_tweets.empty:
         row = {
            'conversation_id': [tweet_data['conversation_id'][i]],
            'id': [tweet_data['id'][i]],
            'public_metrics': [tweet_data['public_metrics'][i]],
            'text': [tweet_data['text'][i]],
         }
         selected_tweets = pd.DataFrame(row, index=[0])
      else:
         selected_tweets = selected_tweets.append(tweet_data.iloc[i], ignore_index = True)
```

Tweets selezionati

```
for i in tweet_data.index:
   # conta se positivo e negativo
   if (b_positive and positive <= 5 ) or (not positive  and negative <= 5):
      conversation = api.request('tweets/search/recent',
                        {'query': ('conversation_id:'+str(tweet_data["conversation_id"][i])),
                         'tweet.fields': "author_id,in_reply_to_user_id,conversation_id",
                         'max_results':"100"
                         }).json()
       # calcolo next_token

      if conversation["meta"]["result_count"]>0:
         for x in conversation["data"]:
            if conversation_data.empty:
               conversation_data = pd.DataFrame(x, dtype=str, index=[0])
            else:
               temp = pd.DataFrame(x, dtype=str, index=[0])
               conversation_data = conversation_data.append(temp, ignore_index=True)
```

Timeline

```
for i in conversation_data.index:
    if count >= init and count <= finish:
        timeline = api.request('users/:%s/tweets' % (conversation_data["author_id"][i]),{
                            'tweet.fields': "author_id",
                             'max_results': "100"}).json()

        if "'detail': 'Could not find user with id:" in str(timeline):
            print("ERROR: user_id not found")
        elif "detail': 'Sorry, you are not authorized to see the user with" in str(timeline):
            print("ERROR: private user_id")
        else:
            # calcolo next_token

            # scarico i dati e li memorizzo del dataframe come visto prima
```

02
PROCESSARE I DATI

Python e R

Pulizia dei dati – conversazioni e timeline

```
for i in conversation_data.index:
    if result.empty:
        aux = {'author_id': [str(conversation_data["author_id"][i])],
               'in_reply_to_user_id': [str(conversation_data["in_reply_to_user_id"][i])],
               'conversation_id': [str(conversation_data["conversation_id"][i])]}
        result = pd.DataFrame(data = aux)
    else:
        aux = {'author_id': str(conversation_data["author_id"][i]),
               'in_reply_to_user_id': str(conversation_data["in_reply_to_user_id"][i]),
               'conversation_id': str(conversation_data["conversation_id"][i])}
        result = result.append(aux, ignore_index = True)
```

```
timeline_data = timeline_data[timeline_data.text.str.contains('greenpass|vaccino|super green|siero')]
```

Arricchimento dati - timeline

```
timeline_data = timeline_data.assign(sentiment = "nan")

for i in timeline_data.index:
   timeline_data["sentiment"][i] = str(sentiment_classifier.predict([timeline_data["text"][i]]))
```

Arricchimento dati – calcolo polarizzazione

```
aux = pd.DataFrame()
for id in timeline_data.groupby(['author_id']).groups.keys():
    positive = timeline_data[(timeline_data.author_id == id) & (timeline_data.sentiment == "['positive']")].count()['sentiment']
    negative = timeline_data[(timeline_data.author_id == id) & (timeline_data.sentiment == "['negative']")].count()['sentiment']
    total = positive + negative
    polar = positive - negative
    if result.empty:
        aux = {'id': [id],
               'polarization': [(polar/total)]}
        result = pd.DataFrame(data = aux)
    else:
        aux = {'id': id,
               'polarization': (polar / total)}
        result = result.append(aux, ignore_index=True)
```

Aggregazione dei dati

```
> conversation = merge(x = conversation, y = selected_tweets, by = "conversation_id", all.x = TRUE)
> conversation$conversation_id <- NULL
> conversation$id <- NULL
> conversation$public_metrics <- NULL
```

03 ANALIZZARE I DATI

R

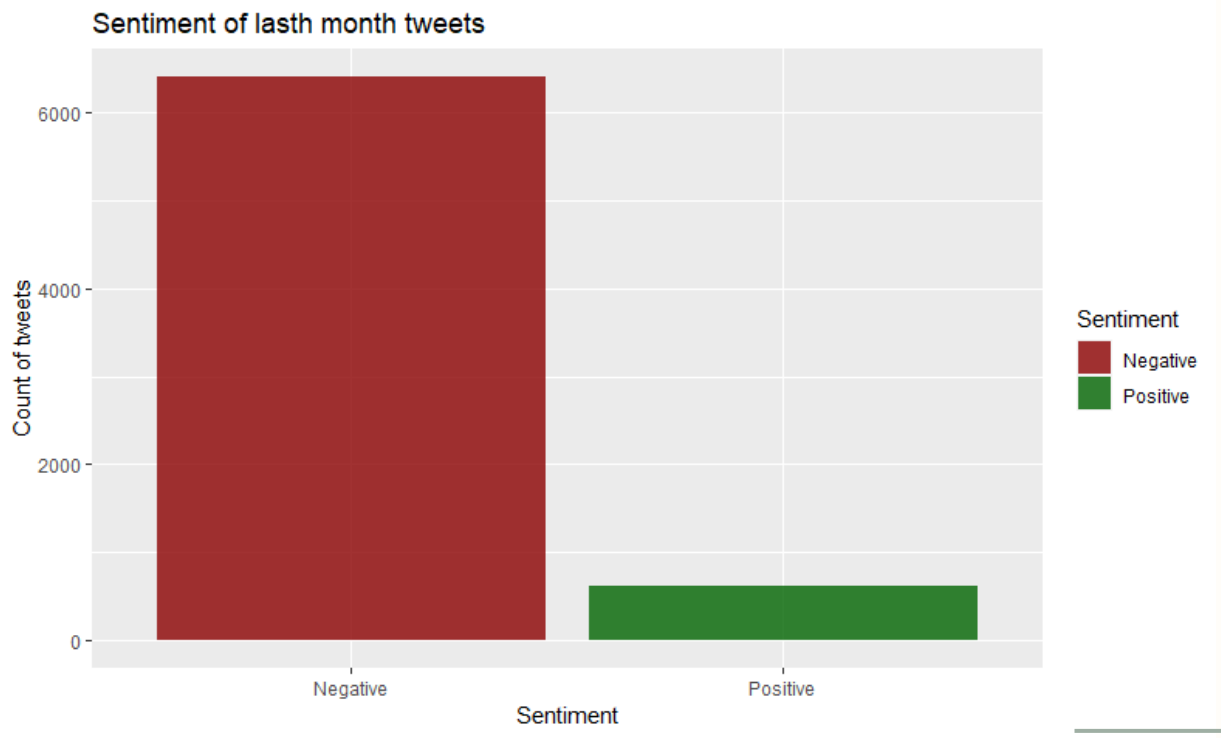
statistiche

```
> count(tweets, "text")
       text         freq
1 ['negative'] 6404
2 ['positive']    625
> nrow(timeline)
[1] 1242
> nrow(conversation)
[1] 1955
> nrow(tweets)
[1] 7029
> nrow(selected_tweets)
[1] 10
```

Grafici - sentiment

```
> ggplot(data = tweets) +
    geom_bar(mapping = aes(x = text, fill = text), alpha = 0.8) +
    labs(x = "Sentiment", y = "Count of tweets", title = "Sentiment of lasth month tweets", fill =
"Sentiment") +
    scale_fill_manual(labels = c("Negative", "Positive"), values = c("dark red", "dark green")) +
    scale_x_discrete(labels= c("Negative", "Positive"))
```

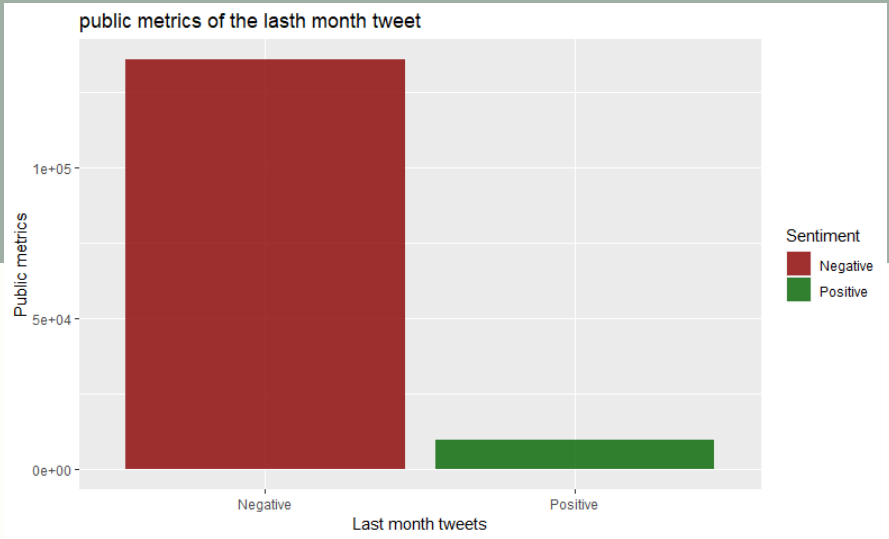
Grafico - sentiment



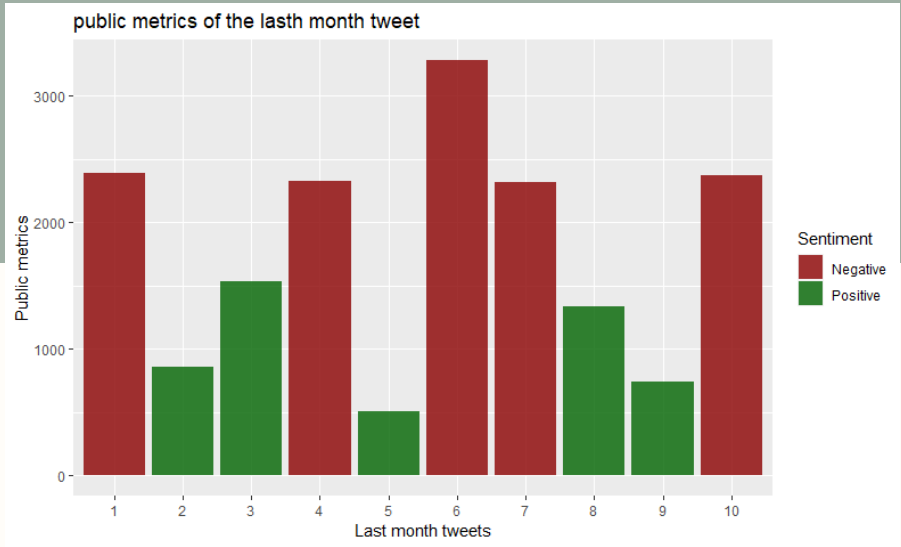
Grafici – sentiment metriche

```
> ggplot(data = tweets, aes(x = text, y=public_metrics, fill = text)) +
  geom_bar(stat = "identity", alpha = 0.8) +
  labs(x = "Last month tweets", y = "Public metrics", title = "public metrics of the lasth month tweet",
fill = "Sentiment") +
  scale_fill_manual(labels = c("Negative", "Positive"), values = c("dark red", "dark green")) +
  scale_x_discrete(labels= c("Negative", "Positive"))
```

Grafici – sentiment metriche



Tutti i tweet

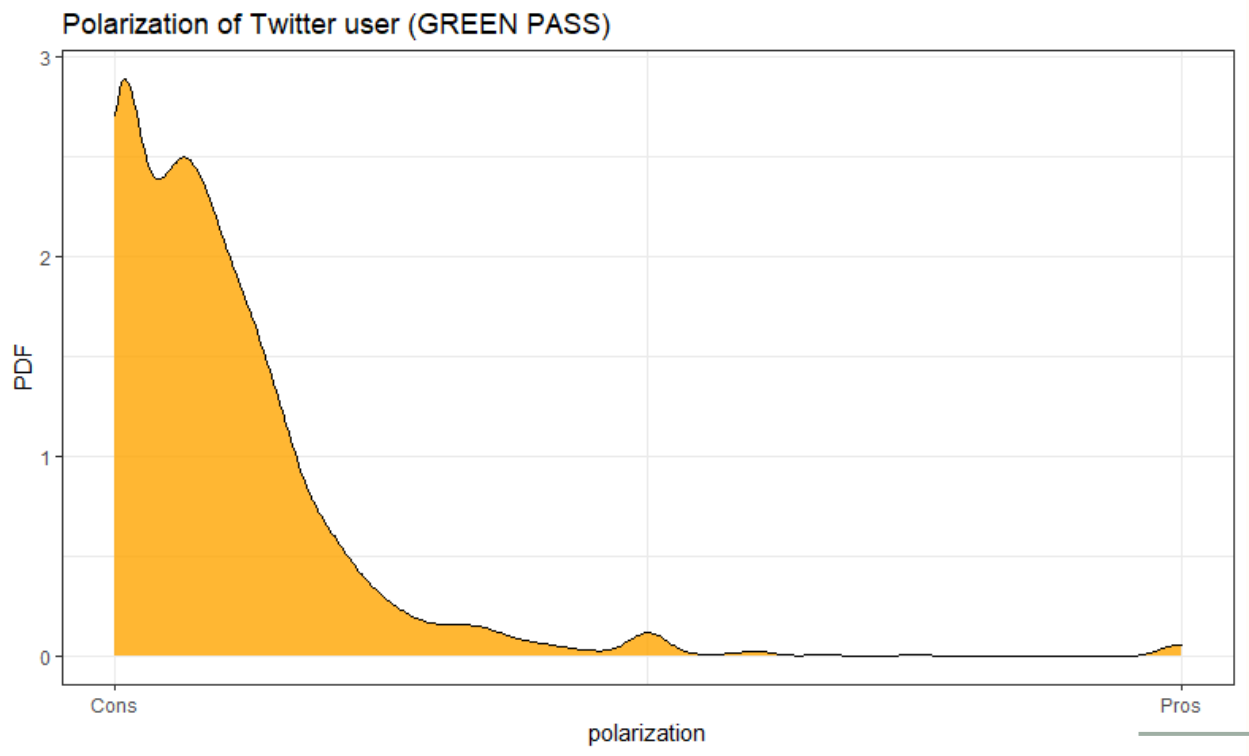


Tweet selezionati

Grafici – polarizzazione

```
> ggplot(timeline, aes(x = polarization)) +
  geom_density(fill="orange", alpha=0.8) +
  scale_x_continuous(limits = c(-1,1),
                     breaks=c(-1,1),
                     labels=c("Cons","Pros")) +
  labs(x = "polarization", y = "PDF", title = "Polarization of Twitter user (GREEN PASS)") +
  theme_bw()
```

Grafico - polarizzazione



Grafo – attributi archi

```
> net <- graph_from_data_frame(d=conversation, directed=T)
> net <- simplify(net, remove.multiple = F, remove.loops = T)

> V(net)$frame.color <- "black"
> V(net)$label <- ""

> ecol <- rep("dark green", ecount(net))
> ecol[E(net)$text=="['negative']"] <- "dark red"
> E(net)$color = ecol
> elty <- rep(1, ecount(net))
> elty[E(net)$text=="['negative']"] <- 2
> E(net)$lty = elty
```

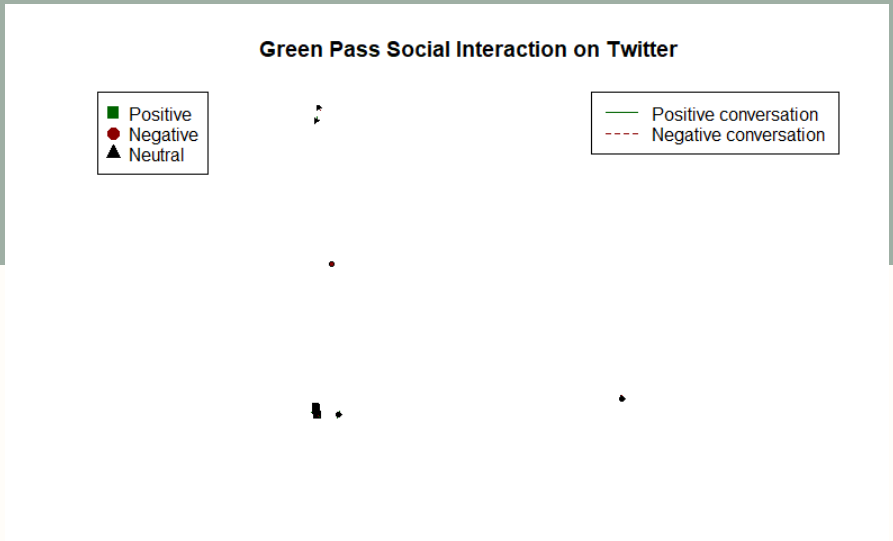
Grafo – attributi vertici

```
> vpol <- rep(0, vcount(net))
> vpol[V(net)$name%in%timeline$id] <- timeline$polarization

> vcol <- rep("black", vcount(net))
> vcol[vpol < 0] <- "dark red"
> vcol[vpol > 0] <- "dark green"
> V(net)$color <- vcol

> vsha <- rep("triangle", vcount(net))
> vsha[vpol < 0] <- "circle"
> vsha[vpol > 0] <- "square"
> V(net)$shape <- vsha

> vpol[vpol < 0] <- vpol[vpol < 0]*-1
> vpol[vpol == 0] <- vpol[vpol == 0]+1
> V(net)$size <- vpol*4
```

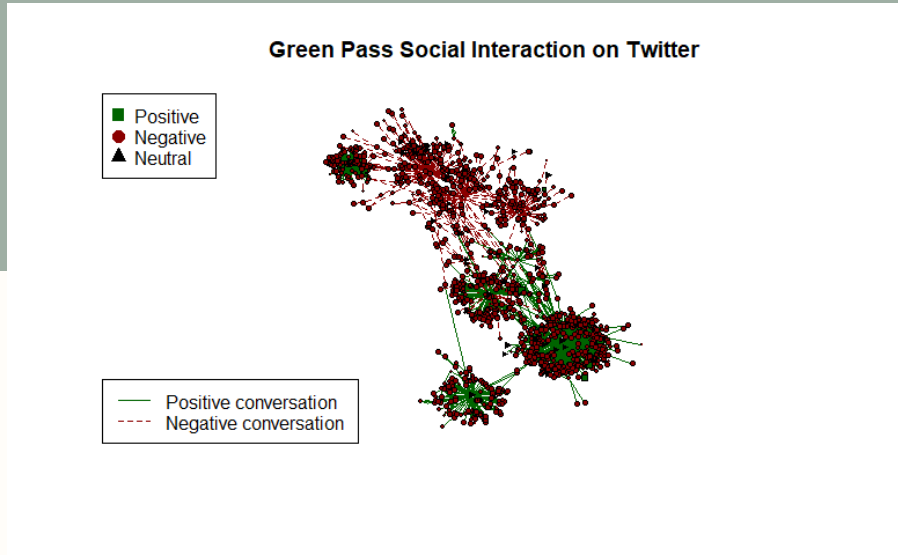
Grafo – plot

```
> l1 <- layout.fruchterman.reingold(net)
> plot(net, layout=l1, edge.arrow.mode=0, main="Green Pass Social Interaction on Twitter")
> legend('topleft',legend=c("Positive", "Negative", "Neutral"), pch=c(15,16,17), pt.cex=1.5, col=c('dark green', 'dark red', 'black'))
> legend('topright',legend=c("Positive conversation", "Negative conversation"), lty=c(1,2), col=c('dark green', 'dark red'))

> cl <- clusters(net)
> net <- induced.subgraph(net, which(cl$membership == which.max(cl$csize)))
```

Grafici

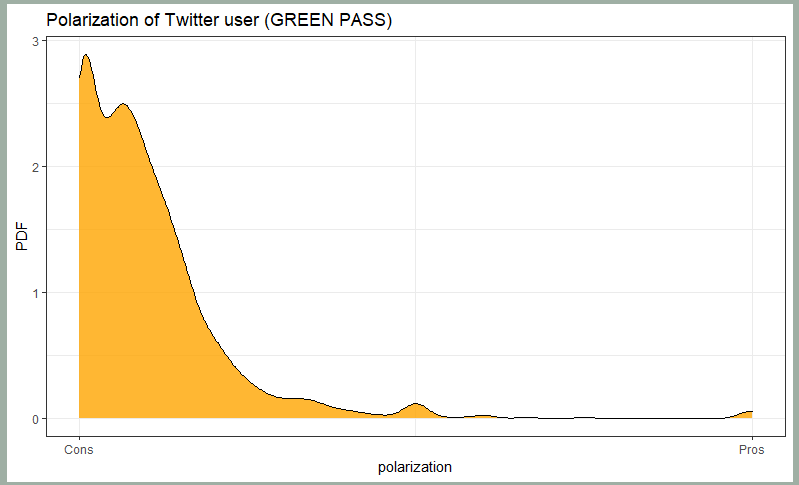


Sconnesso



Componente gigante

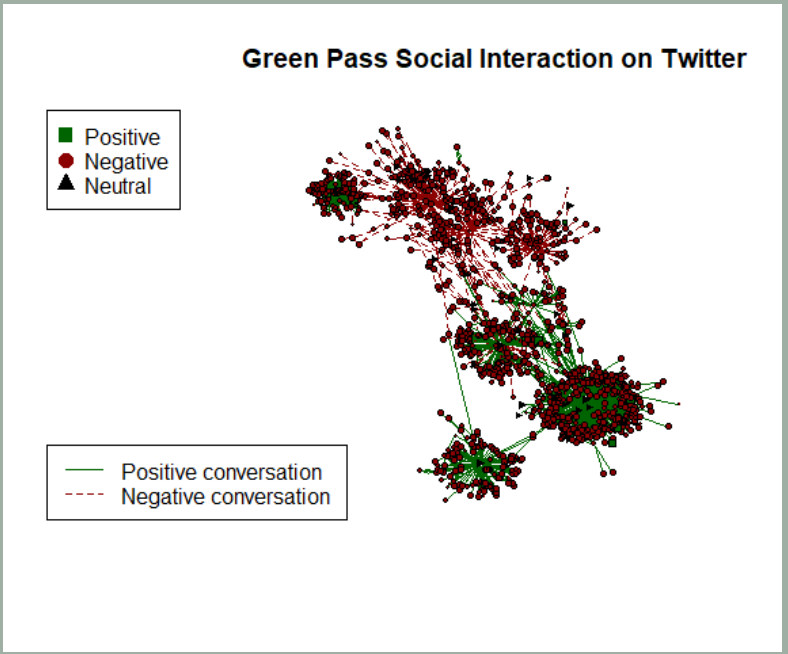
04
VALUTARE I DATI



Polarizzazione

Utenti di Twitter fortemente polarizzati solo negativamente: maggioranza di utenti contrari alle disposizioni degli ultimi decreti-legge.

Italiani per lo più persone contrario agli ultimi provvedimenti governativi.



Echo chambers

Concentrazione intorno tweet “positivi”.

Una “camera d’eco” → scarsa presenza di profili favorevoli impedisce creazione ambiente in cui la loro idea sia condivisa.

Nel centro vertici fortemente polarizzati.

Vertici isolati meno polarizzati.
Ponti tra i vari cluster: polarizzati negativamente

THANKS!

879607@stud.unive.it
Ca’ Foscary University, Venice
Italy