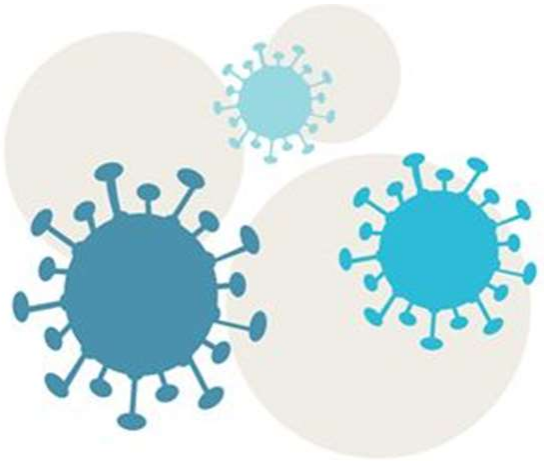
Twitter Polarization and Vaccines



- *Armanini Justin – 830103*
- Caiffa Emanuele – 872515
- *Palomba Eleonora – 876479*
- *Zayeva Nataliya – 867981*

Laurea Magistrale in
DATASCIENCE

UNIVERSITÀ DEGLI STUDI DI MILANO
BICOCCA

INTRODUZIONE

- ***Background***: per tutto il 2021, il tema delle vaccinazioni contro il Covid-19 ha occupato un ruolo di rilievo all'interno del dibattito pubblico
- ***Obiettivi***: il progetto qui presentato si pone le seguenti domande:

  - Chi sono gli italiani più *influenti* in materia di vaccini?

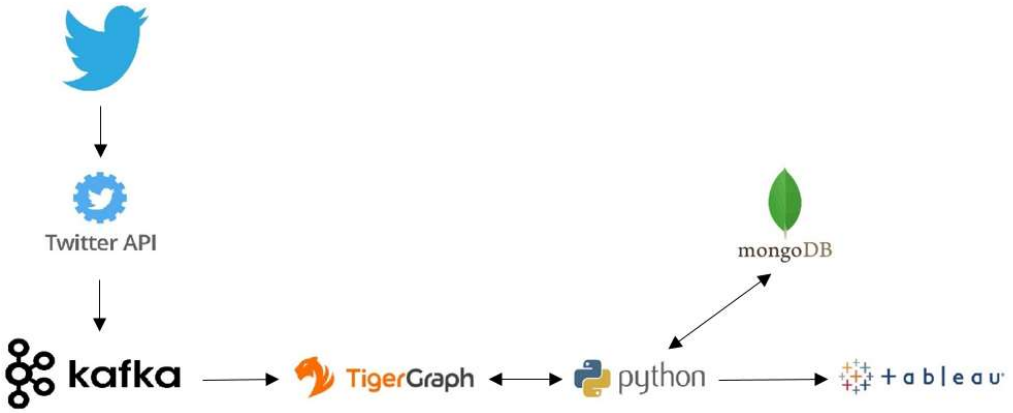
  - Qual è il lessico più frequente quando si parla di vaccini?

  - E' possibile osservare una *polarizzazione* del dibattito? Esistono comunità di pro-vax e no-vax che restano ancorate alle loro posizioni rifiutando il confronto con idee contrastanti?

- ***Metodo***: le due V affrontate sono state di V di *Velocità* e V di *Volume*. Sono stati analizzati i tweet di utenti italiani tra Febbraio e Maggio 2021. E' stato applicato un algoritmo di community detection per individuare le comunità pro-vax e no-vax tenendo conto delle interazioni reciproche degli utenti.

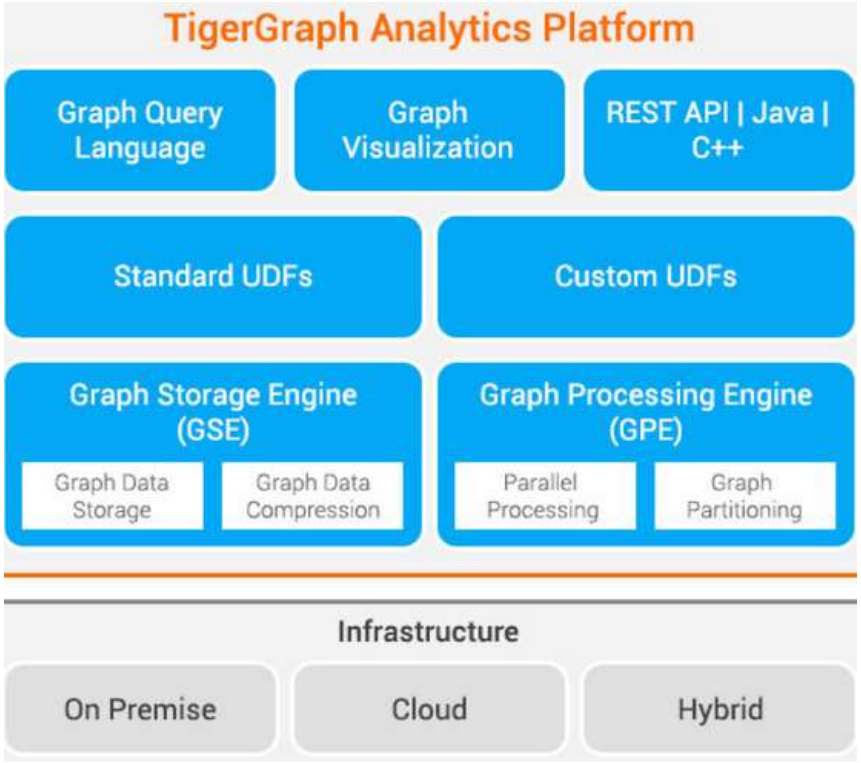
ARCHITETTURA

- Twitter API: raccolta dati in streaming tramite le API di Twitter. Questo ha consentito una raccolta dei dati in streaming, realizzando così la V della velocità.
- Apache Kafka: disaccoppia la fase di memorizzazione dalla fase di acquisizione tramite *publish and subscribe*
- TigerGraph: DBMS a grafo distruibito con engine nativo. Il linguaggio utilizzato è GSQL, una sintassi di SQL-like
- MongoDB: DBMS Documentale, caratterizzato da strutture annidate e NoSQL like



TigerGraph

- DBMS a grafo distribuito con storage engine nativo
- Architettura interna prevede un Graph Storage Engine (GSE) e Graph Process Engine (GPE), che fornisce una funzionalità di partitioning dei dati in maniera automatica su più nodi, realizzando così lo scaling-out della soluzione al bisogno
- Unico svantaggio: schema dei dati non *free*, ma deve essere definito all'inizio in fase di progettazione



Configurazione DBMS - TigerGraph

- La configurazione del DBMS è stata effettuata tramite GraphStudio, un'interfaccia grafica che permette di interagire in maniera semplice e intuitiva.
- È stata utilizzata una versione di TigerGraph implementata in cloud su una macchina disponibile tramite il servizio AWS e sfruttando:
  - → 2 nodi *Partition* per la distribuzione orizzontale;
  - → 2 nodi *Replication* per la duplicazione dei dati.

  In questo modo è stata aumentata sia l'affidabilità tramite la replica dei dati, sia le performance aumentando il grado di parallelismo.
- La fase di acquisizione ha fatto uso di Tigergraph "centralizzato"; tuttavia, questo non ha richiesto alcuna modifica dal punto di vista implementativo, in quanto le query vengono eseguite nello stesso modo

MongoDB

- DBMS Documentale, non completamente open-source, che supporta i file in formato JSON
- È stato utilizzato il metodo “sharding” per gestire la scalabilità orizzontale dei dati. Trattasi di un metodo che, attraverso l’aggiunta di nodi, permette al sistema di distribuire sia i dati che il carico applicativo
- Lo sharded cluster, ovvero il risultato dello sharding consiste nelle seguenti componenti:

1. *shards (o replica sets)*
2. *routers (o mongos)*
3. *config server*

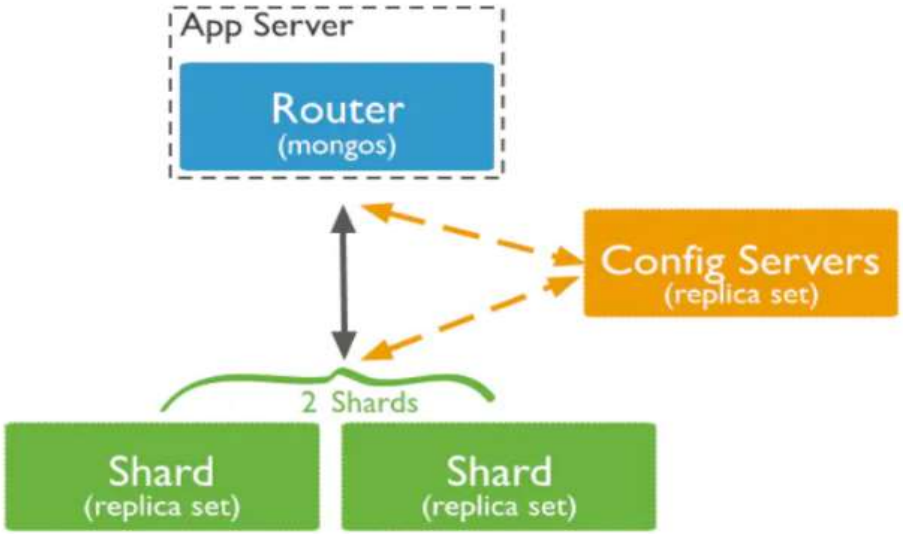
- *I documenti sono distribuiti attraverso gli shards grazie alle chiavi, o “shard keys”, e sono organizzati in “chunks”.*

MongoDB in Sharding

Il sistema utilizzato è stato configurato con:

- due *shard cluster* in replica set, ciascuno con tre nodi *mongod;*
- un *config server* in replica set con tre nodi;
- un *router* per gestire il tutto.

Tale configurazione è stata scelta per gestire al meglio il trade-off fra risorse a disposizione e scalabilità orizzontale.

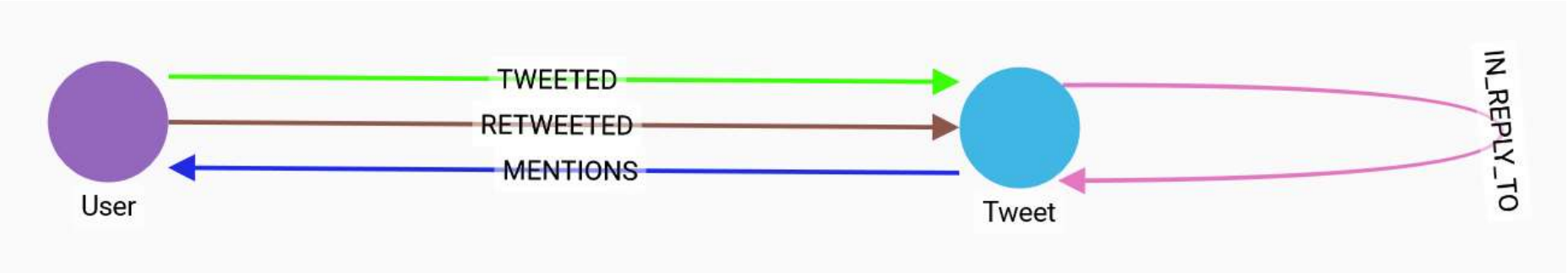


Dati

- I dati utilizzati si riferiscono ai tweet e agli utenti attivi sulla piattaforma nel periodo che va dal 22 Febbraio 2021 al 23 Maggio 2021.

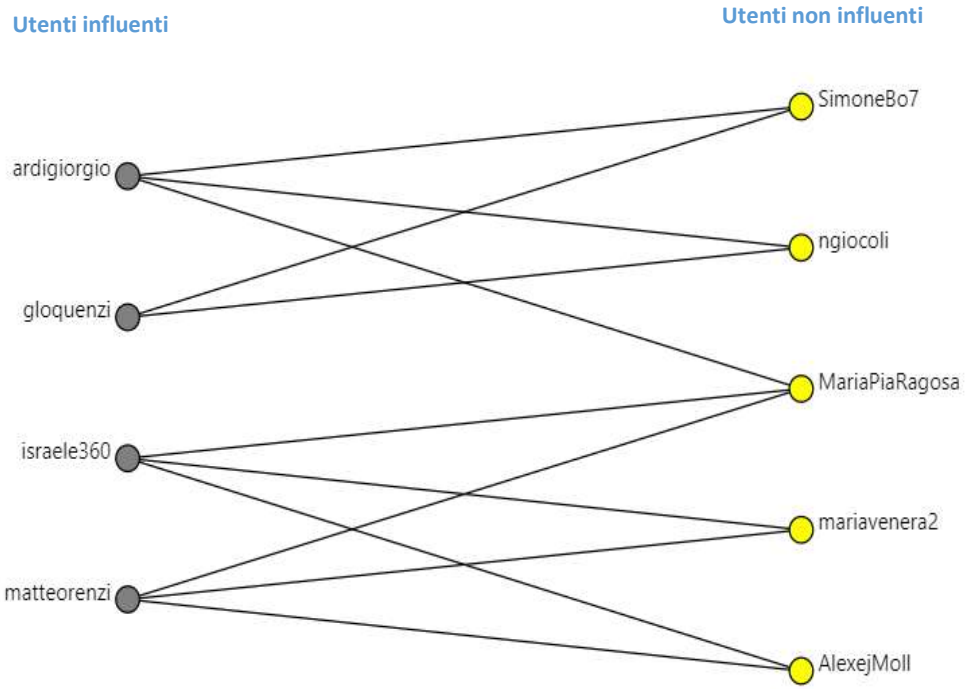
- Tweet sono stati scaricati tramite API in formato JSON

- Le informazioni estratte dai tweet sono state memorizzate su Tigergraph secondo lo schema seguente:



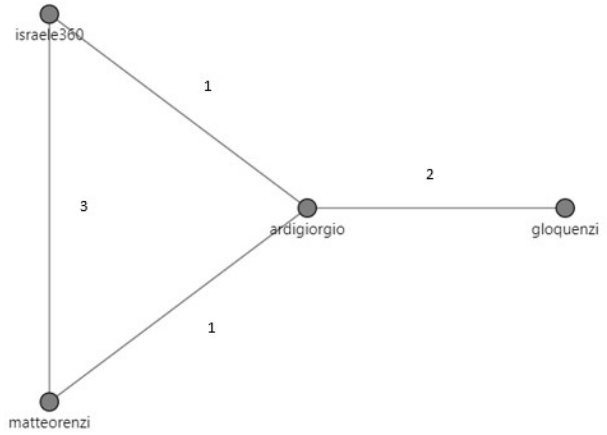
Analisi e Risultati

- L'obiettivo principale delle analisi è stato esattamente quello di individuare e quantificare, se presente, la polarizzazione su Twitter.
- Sono state individuate delle metriche che sono utili per distinguere gli utenti influenti e non influenti
- Successivamente, per ciascuna settimana e per ciascun utente è stata calcolata la media delle metriche complesse. Infine, avendo ordinato gli utenti per tale media, sono stati classificati come utenti influenti quelli aventi media superiore al 95-esimo percentile.
- La divisione in utenti influenti e non ha permesso di realizzare il grafo bipartito che mette in relazione utenti influenti con quelli non influenti che hanno retwittato i contenuti dei primi

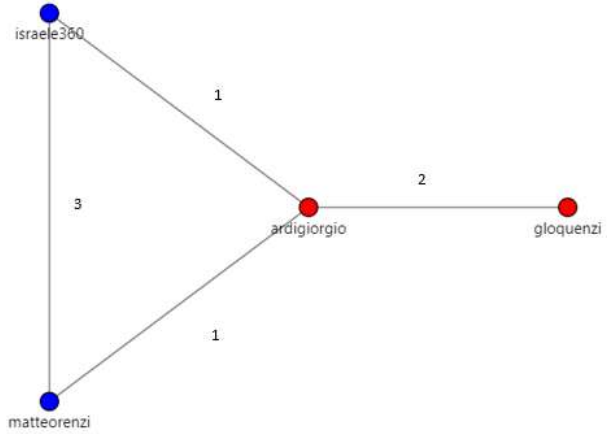


Analisi e Risultati

- Da questo grafo è stato ottenuto il grafo di “proiezione”, ovvero un grafo i cui nodi sono soltanto gli utenti influenti e gli archi pesati con il numero di utenti non influenti che i due nodi collegati condividono nel grafo bipartito.

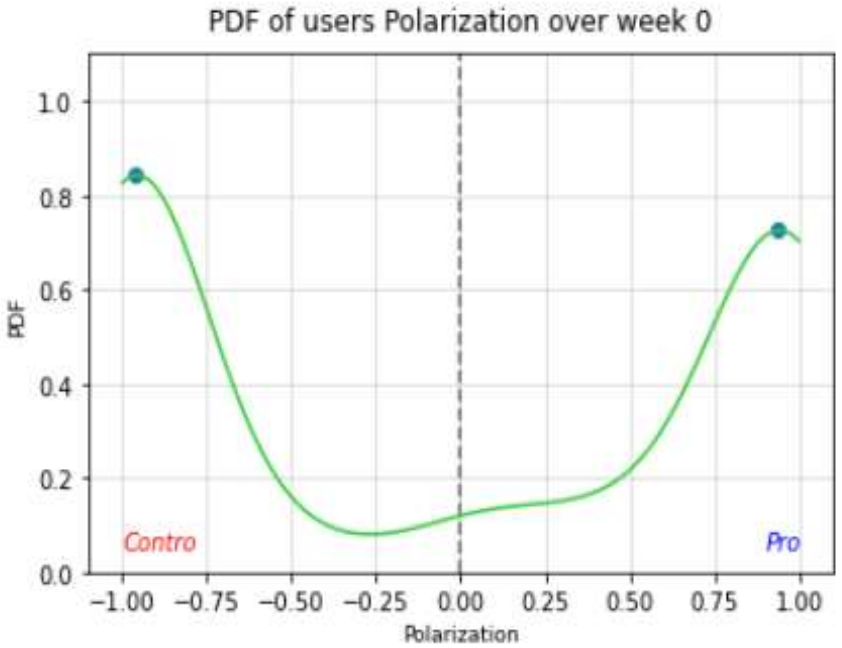


- È stato quindi applicato l’algoritmo di clustering Louvain tramite la libreria scikit-network implementata in Python.
- I risultati hanno messo in evidenza l’esistenza di due grandi cluster in quasi tutte le settimane.



Analisi e Risultati

- Per ciascun cluster è stato estratto un campione casuale di utenti influenti per i quali sono stati valutati manualmente i relativi tweet e retweet. Quest'analisi ha riscontrato che, pur non essendo perfetti, i cluster ottenuti coincidevano con le due community che ci si aspettava esistessero: pro-vax e no-vax.
- A questo punto è stato possibile, per ciascun utente non influente (che rappresentano per ciascuna settimana il 95% della totalità degli utenti) calcolarne la polarizzazione
- Polarizzazione di u: $p(u) = \frac{x-y}{x+y}$
- Tale calcolo di polarizzazione è stato applicato soltanto agli utenti che avessero fatto >= 10 retweet nella settimana di interesse
- Infine tramite Kernel Density Estimation si è ottenuta una stima della funzione di densità di probabilità



Analisi e Risultati

- Per quanto riguarda la seconda domanda di ricerca, invece, sono stati estratti tutti i termini presenti all'interno di ognuno dei tweet raccolti e si è provveduto a sottoporli ad un processo di normalizzazione dei lemmi

- Sono state utilizzate due modalità operative per creare la lista di 140 termini che potenzialmente avrebbero potuto ottenere frequenze elevate:
  - → Consultazione di siti web
  - → Lettura periodica dei post twittati dagli utenti italiani che contenevano l'hashtag "#vaccino"

Conclusioni

- Gli utenti più influenti con la polarità positiva e negativa sono risultati Roberto Burioni e Claudio Borghi. Per quanto riguarda gli utenti neutrali non è stato possibile individuare un unico utente più influente degli altri.
- I termini utilizzati più di frequente durante le prime settimane sono risultati “morte” e “AstraZeneca” , e questo potrebbe essere dovuto alla registrazione dei primi casi di trombosi avvenuti a marzo. Nei mesi successivi, le parole più utilizzate risultano invece essere “Pfizer”, “AstraZeneca” e “prenotazione”.
- Si è osservato che la polarizzazione sia particolarmente accentuata, ovvero che la maggior parte degli utenti sia ancorata ad una delle due posizioni. La numerosità della community dei “pro” e dei “contro” risulta molto simile durante le prime settimane. Nelle ultime settimane è possibile osservare un graduale, ma costante aumento della numerosità della community dei “contro”, fino a raggiungere il picco della densità di probabilità nella 20esima settimana.

Grazie per l’attenzione!