



UNIVERSITÀ DI PISA

*Master Universitario di I Livello in Cybersecurity*



# Corso CyberIntelligence: esercitazione su ES e Kibana

Tiziano Fagni

IIT,CNR

[tiziano.fagni@iit.cnr.it](mailto:tiziano.fagni@iit.cnr.it)



# Lezione del 11/6/2022

# Esercizio 1

## Obiettivo

Sfruttando il template presente sul file “TwitterMappingFromScript.json” in “cint/esercitazione/elastic/mapping\_2.0”, creare un nuovo mapping per ES che consideri anche i campi di arricchimento definiti negli Esercizi 4 e 5 su NiFi. I campi nuovi fanno riferimento a questi dati:

```
"politicians": {  
  "polarity": "-1",  
  "names": ["Salvini"]  
},  
"webpage": {  
  "url": "https://t.co/6wixF7WHyq",  
  "title": "PesNew Era su Twitter...",  
  "content": "Ora sei..."  
}
```

# Esercizi su query

Trovate la documentazione di Elasticsearch sulle query qui:

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html>

Tipi di query che vedremo:

- Full text queries
- Term level queries

Faremo le nostre query sull'indice "twitter\_data" creato dal dataflow "TwitterForQueries".

# Full text queries: “match” query

```
GET /<index-name>/_search
{
  "from" : 0,
  "size" : 100,
  "query": {
    "match" : {
      "message" : {
        "query" : "this is a test",
        "operator" : "and"
      }
    }
  }
}
```

Cerca le parole specificate dalla keyword “query” nel campo padre. Posso dire se cercarle in “and” oppure “or” tramite la keyword “operator”.

# Esercizio

Sfruttando la “match” query, cercare tutti i tweet che verificano le query:

1. frase “putin guerra” con parole in “and” sul campo contenente il tweet.
2. frase “conte governo” con parole in “or” sul campo contenente il tweet.
3. frase “covid vaccino” con parole in “and” sul campo che fa riferimento il contenuto della pagina Web collegata.

# Full text queries: “match phrase” query

GET /\_search

```
{
  "query": {
    "match_phrase" : {
      "message" : {
        "query": "this is a test",
        "slop": 0
      }
    }
  }
}
```

Cerca il matching per la frase specificata nel campo indicato. Con il parametro “slop” si può indicare quanto preciso può essere fatto il matching considerando l’ordine delle parole.

**Il parametro “slop” cerca di rispondere a questa domanda**

“By how far apart we mean how many times do you need to move a term in order to make the query and document match?”

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-match-query-phrase.html>

<https://www.elastic.co/guide/en/elasticsearch/guide/current/slop.html>

# Esercizio

Sfruttando la “match phrase” query e usando un campo a vostra scelta, cercare tutti i tweet che verificano le query:

1. “pd draghi” con vari valori di “slop”.
2. “salvini lega” con vari valori di “slop”.



# Full text queries: “match phrase prefix” query

```
GET /_search
{
  "query": {
    "match_phrase_prefix" : {
      "message": {
        "query" : "this is a t",
        "slop": 0,
        "max_expansions" : 50
      }
    }
  }
}
```

Cerca il matching nel campo indicato per la frase specificata facendo l'espansione dell'ultimo termine.

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-match-query-phrase-prefix.html>

# Esercizio

Sfruttando la “match phrase prefix” query e usando un campo a vostra scelta, cercare tutti i tweet che verificano le query:

1. “conte gov” con vari valori di “slop”.
2. “governo c” con vari valori di “slop”.

E' possibile utilizzare sia la scheda “Dev tools” sia la scheda “Discover”

# Full text queries: Multi Match query

```
GET /_search
{
  "query": {
    "multi_match" : {
      "query": "brown fox",
      "type": "best_fields",
      "fields": [ "subject^3", "message" ],
    }
  }
}
```

Stessa semantica della  
“match” query ma con la  
possibilità di utilizzare più  
campi di ricerca e  
combinare gli score.

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-multi-match-query.html>

# Esercizio

Sfruttando la “multi match” query, provate a cercare qualcosa sui campi “text” e “webpage.content”

- assegnando 3 volte più importanza a quello che trovate in “webpage.content”.
- provando a combinare lo score finale sfruttando le diverse policy disponibili (vedi <https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-multi-match-query.html>).

# Full text queries: “query string” query

Si usa il “**query string format**” per formulare le query

GET /\_search

```
{  
  "query": {  
    "query_string" : {  
      "default_field" : "content",  
      "query" : "(this AND that) OR thus",  
      "default_operator" : "OR"  
    }  
  }  
}
```

Molti altri parametri disponibili, vedere

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-query-string-query.html>

# Query string format

- where the `status` field contains `active`

```
status:active
```

- where the `title` field contains `quick` or `brown`. If you omit the OR operator the default operator will be used

```
title:(quick OR brown)  
title:(quick brown)
```

- where the `author` field contains the exact phrase `"john smith"`

```
author:"John Smith"
```

- where any of the fields `book.title`, `book.content` or `book.date` contains `quick` or `brown` (note how we need to escape the `*` with a backslash):

```
book.\*:(quick brown)
```

- where the field `title` has any non-null value:

```
_exists_:title
```

# Query string format (2)

- All days in 2012:

```
date:[2012-01-01 TO 2012-12-31]
```

- Numbers 1..5

```
count:[1 TO 5]
```

- Tags between alpha and omega, excluding alpha and omega:

```
tag:{alpha TO omega}
```

- Numbers from 10 upwards

```
count:[10 TO *]
```

- Dates before 2012

```
date:{* TO 2012-01-01}
```

# Query string format (3)

Ranges with one side unbounded can use the following syntax:

```
age:>10  
age:>=10  
age:<10  
age:<=10
```



To combine an upper and lower bound with the simplified syntax, you would need to join two clauses with an `AND` operator:

```
age:(>=10 AND <20)  
age:(+>=10 +<20)
```

```
quick brown +fox -news
```

states that:

- `fox` must be present
- `news` must not be present
- `quick` and `brown` are optional — their presence increases the relevance



# Query string format (4)

Multiple terms or clauses can be grouped together with parentheses, to form sub-queries:

```
(quick OR brown) AND fox
```

Groups can be used to target a particular field, or to boost the result of a sub-query:

```
status:(active OR pending) title:(full text search)^2
```

E' possibile fare molte altra cose come ricerche fuzzy, di prossimità, con boosting, ecc.!

La documentazione completa è disponibile qui:

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-query-string-query.html#query-string-syntax>

# Esercizio

Dal tab “Discover” di Kibana provare a recuperare i tweet che matchano le condizioni:

1. Autore del tweet con numero di follower tra 50 e 500.
2. Testo del tweet che contiene “russia o putin” e che il contenuto della pagina Web contiene “gas e europa”.
3. Descrizione dell’utente che contiene qualsiasi prefisso “euro” e numero di tweet prodotti compresi tra 500 e 1000

# Term queries

The `term` query finds documents that contain the **exact** term specified in the inverted index. For instance:

```
POST _search
{
  "query": {
    "term": { "user" : "Kimchy" } ❶
  }
}
```

Filters documents that have fields that match any of the provided terms (**not analyzed**). For example:

```
GET /_search
{
  "query": {
    "terms": { "user" : ["kimchy", "elasticsearch"]}
  }
}
```

## Term queries (2)

```
GET _search
{
  "query": {
    "range" : {
      "age" : {
        "gte" : 10,
        "lte" : 20,
        "boost" : 2.0
      }
    }
  }
}
```

```
GET _search
{
  "query": {
    "range" : {
      "date" : {
        "gte" : "now-1d/d",
        "lt" : "now/d"
      }
    }
  }
}
```

<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-range-query.html>

## Altri tipi di term queries utili

- [exists query](#): verifica la presenza di un campo in un documento
- [fuzzy query](#): cerca documenti con termini simili a quello specificato in query sfruttando similarità basata su [Levenshtein](#) edit distance
- [ids query](#): ritorna tutti i documenti che matchano gli ID specificati
- [regexp query](#): ritorna tutti i documenti che matchano una regexp

# Per tutto il resto...

## - Query DSL

Query and filter context

Match All Query

+ Full text queries

+ Term level queries

+ Compound queries

+ Joining queries

+ Geo queries

+ Specialized queries

+ Span queries

Minimum Should Match

Multi Term Query Rewrite

C'è la documentazione!

Guardate a partire da  
<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl.html>

# Query salvate

Dal tab “Discover” è possibile creare viste tabellari specifiche che visualizzano i dati con eventuali filtri applicati.

- Creare una vista che riporta “nome autore”, “testo del tweet” e “numero di follower”
- Creare una vista che riporta “nome autore”, “testo del tweet” e “titolo pagina Web esterna” sui tweet prefiltrati che contengono sicuramente un link a un articolo esterno.



# Lezione del 18/6/2022



# Visualizzazione 1

## Componente “Tag Cloud”

Visualizzare le 10 città più popolari da cui provengono i tweet (vedi campo “place.name”).



## Visualizzazione 1 - bis

Simile al precedente ma le città sono ordinate per media di “friend count”

- Si aggregano i dati per place.name
- Si usa come metrica di ordinamento la media di user.friends\_count

## Visualizzazione 2

## Componente “Tag Cloud”

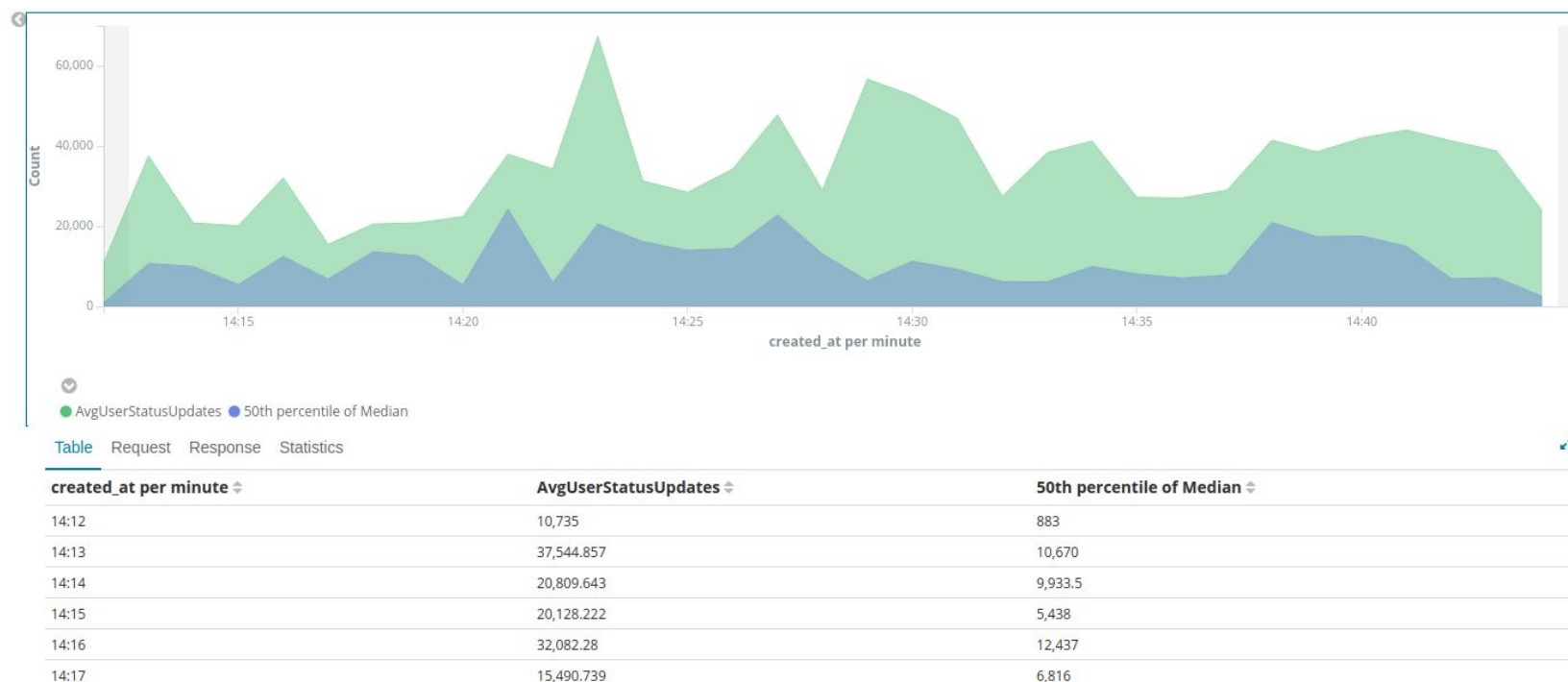
Visualizzare i 50 termini più popolari del “testo pulito” dei tweet.  
Vedere anche come cambia usando il campo raw “text”.



# Visualizzazione 3

## Componente “Vertical Bar”

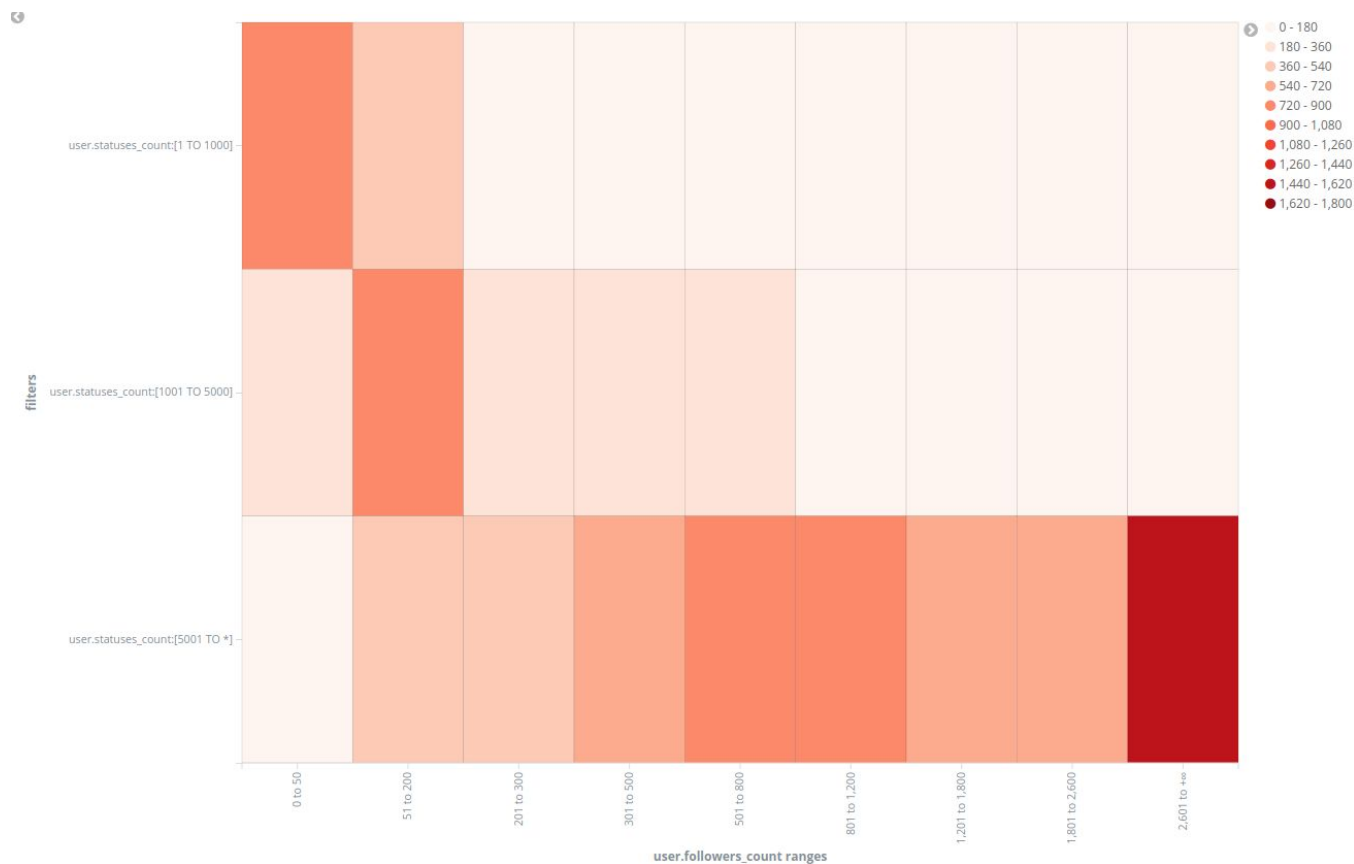
Visualizzare a intervalli di 1 minuto la media e la mediana del numero di aggiornamenti di stato degli utenti che twittano.



# Visualizzazione 4

## Componente “Heat Map”

Realizzare una heatmap che metta in relazione il campo “user.followers\_count” con il campo “user.statuses\_count”.





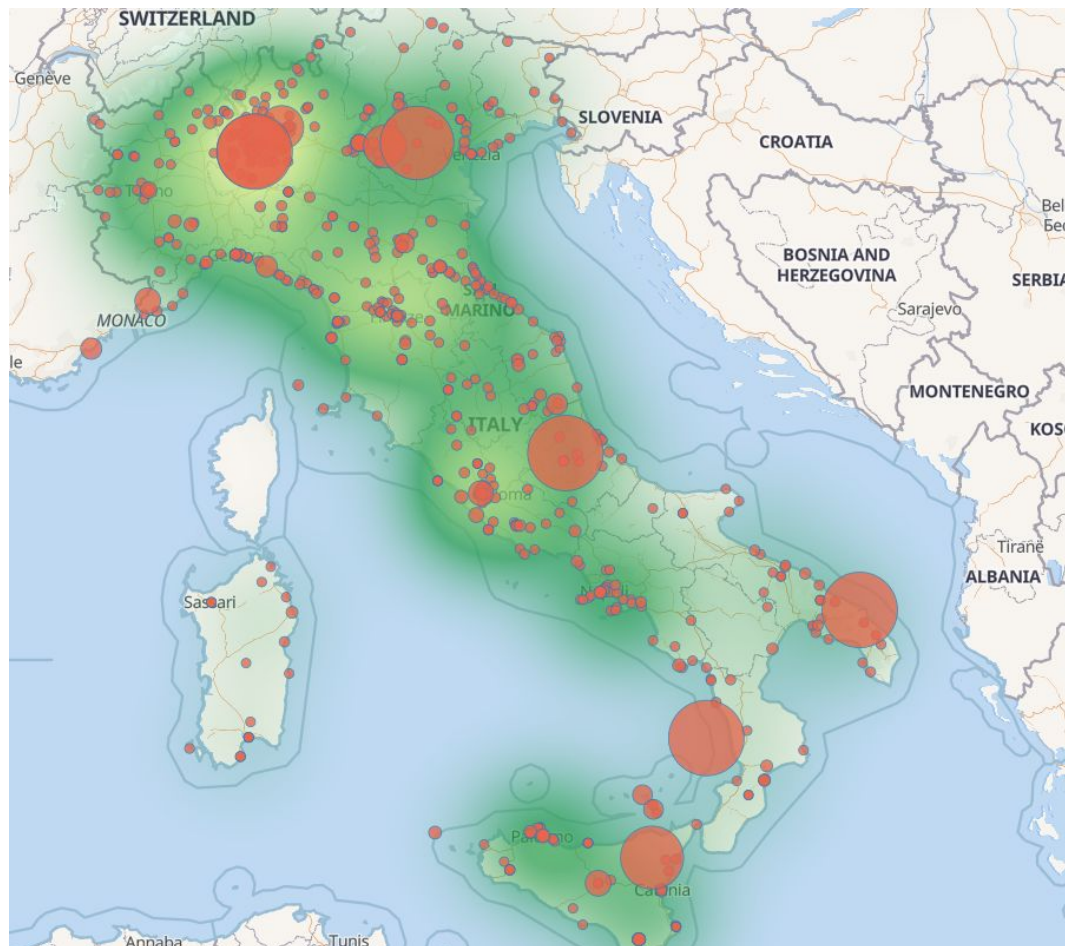
# Visualizzazione 6

## Componente “Map”

Visualizzare i tweet per luogo di provenienza dell'aggiornamento di stato secondo le coordinate GPS riportate (campo “coordinates.coordinates”).

Visualizzare:

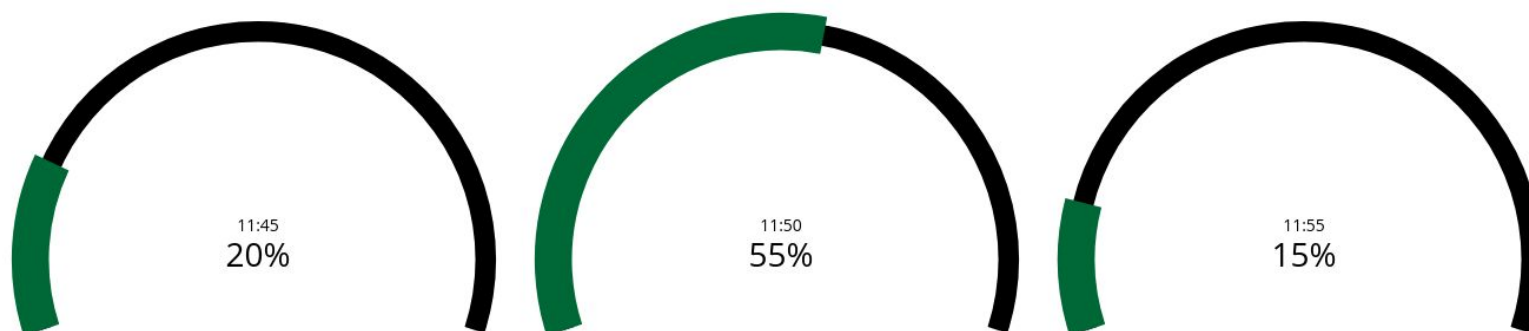
- Popolarità in base al conteggio tweet
- Popolarità in base a numero di follower
- Heatmap distribuzione tweet



# Visualizzazione 7

## Componente “Goal”

Monitorare negli ultimi 15 minuti l’obiettivo di avere almeno “n” immagini distinte postate ogni 5 minuti.

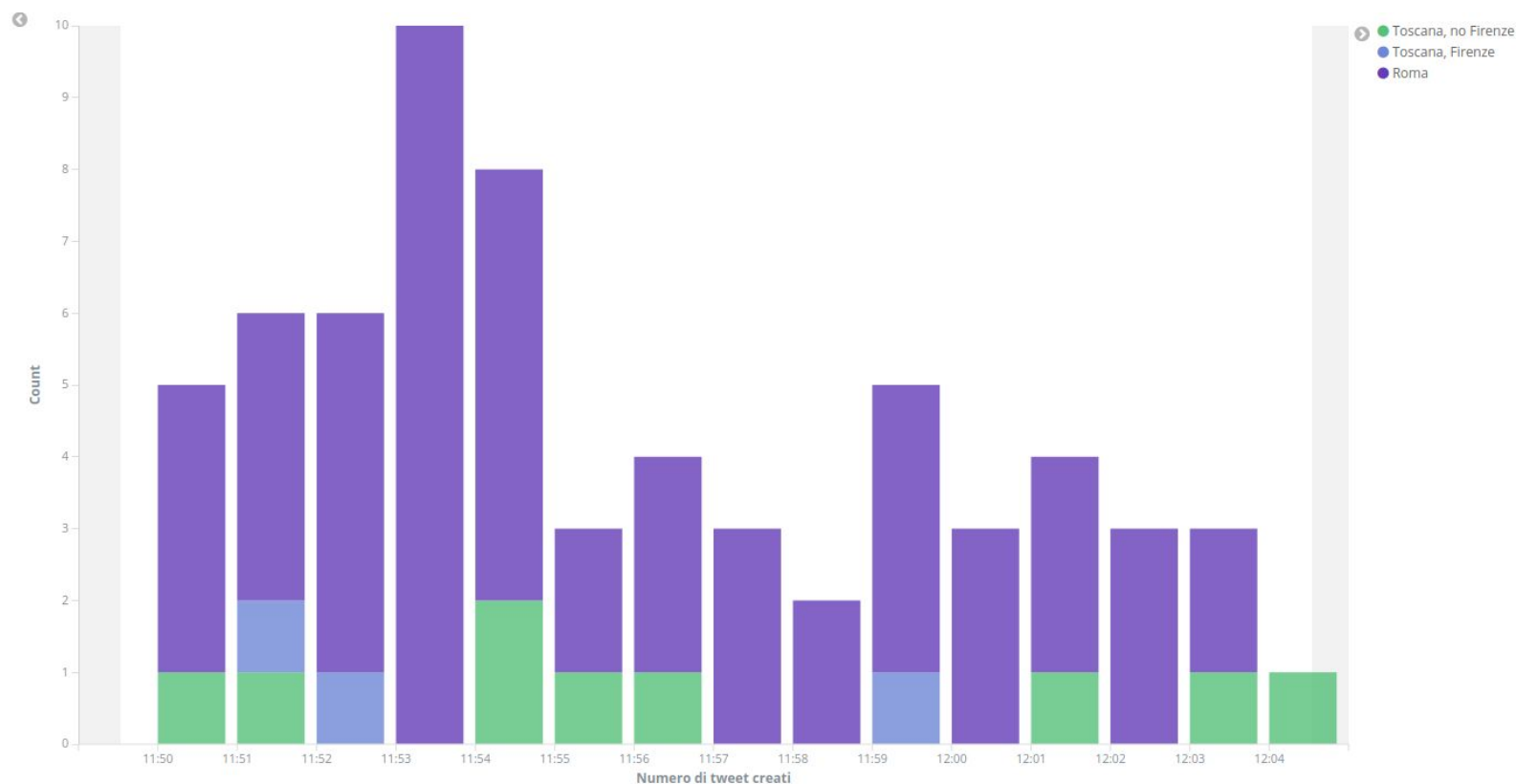




# Visualizzazione 8

## Componente “Vertical Bar”

Visualizzare negli ultimi 15 minuti i tweet arrivati comparando quelli che arrivano dalla Toscana ma non da Firenze, da Firenze oppure da Roma.



# Visualizzazione 9

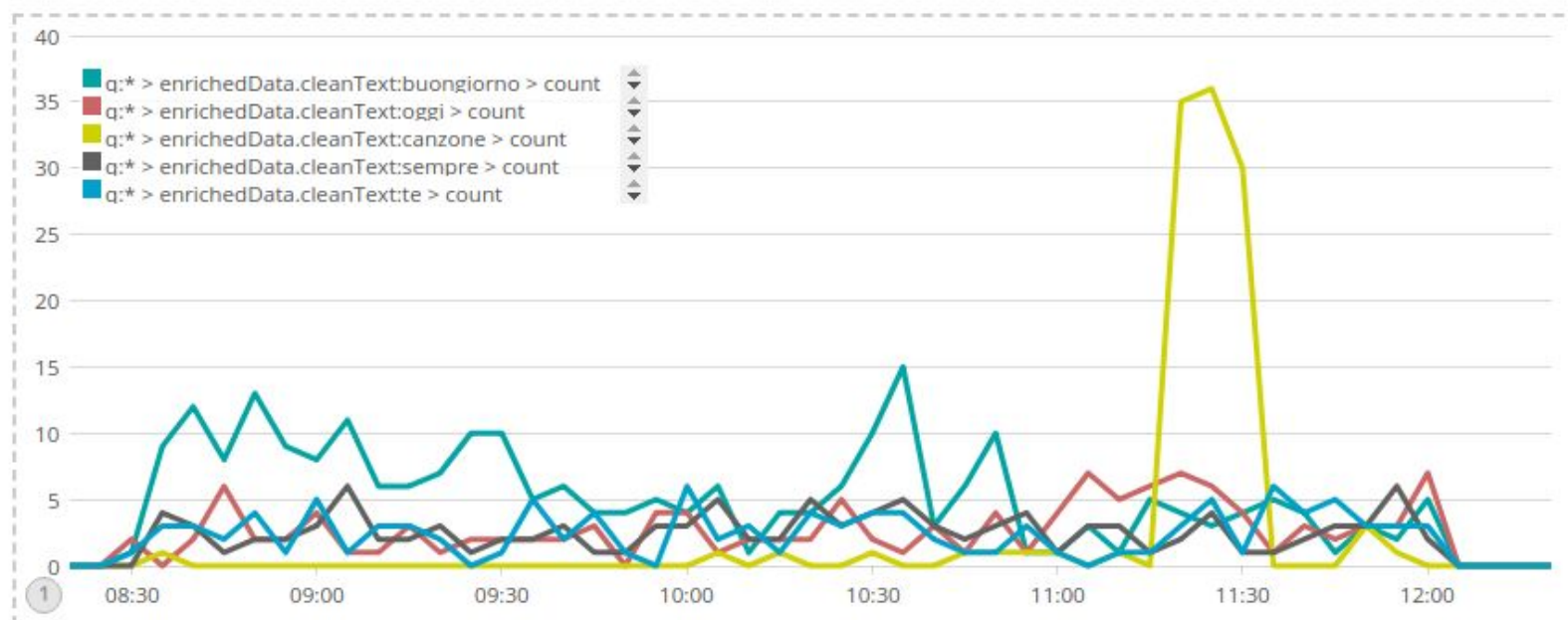
## Componente “Pie”

Visualizzare negli ultimi 15 m come si distribuiscono su una torta gli utenti “non popolari”, “normali” e “popolari”. Definire range opportuni su “user.followers\_count”.



# Timelion: visualizzare velocemente time series

```
.es(q=*,index=worldcup,timefield=created_at,split=enrichedData.cleanText:5),
```

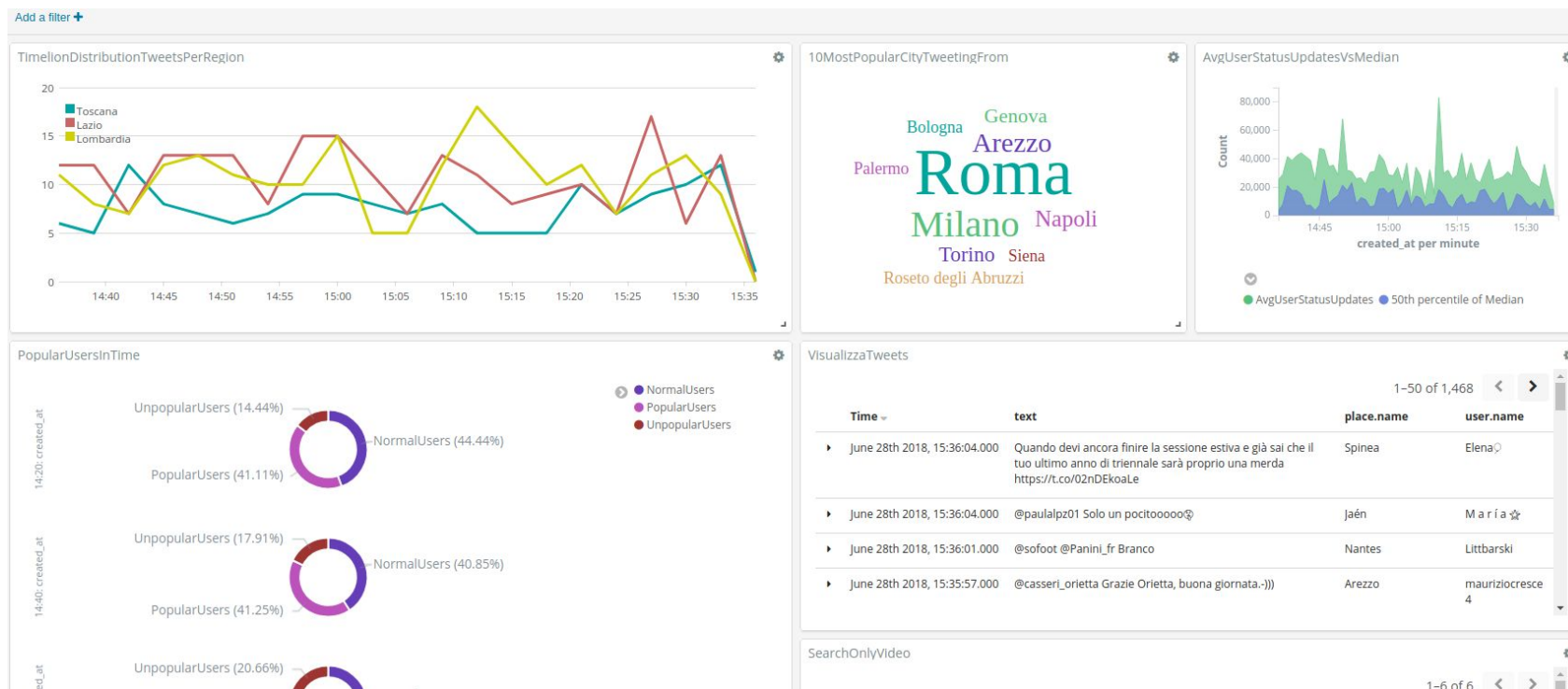


Documentazione: <https://www.elastic.co/guide/en/kibana/current/timelion.html>

Tutorial: <https://www.elastic.co/blog/timelion-tutorial-from-zero-to-hero>

# Adesso costruiamo una bella dashboard...

Potete usare *ricerche*, *visualizzazioni* e *time series*  
*Timelion* salvate





Ok, questo è tutto che che vi serve per fare il vostro bel progettone finale!

Se avete dubbi o domande chiedete pure! :-)

## Esercizio 2

Sfruttando il dataflow NiFi definito nell'Esercizio6ForES di Nifi, verificate come cambiano i dati memorizzati in ES, cambiando nel mapping il valore “dynamic” tra “false”, “true” e “strict”.

Traccia:

- Cancellare eventuale indice esistente.
- Impostare il valore di “dynamic” desiderato nel file di mapping e creare il nuovo indice.
- Caricare qualche dato su ES sfruttando il dataflow di Esercizio6ForES.

<https://www.elastic.co/guide/en/elasticsearch/reference/current/dynamic.html>

## Prima di passare alle visualizzazioni Kibana...

- Creare un nuovo indice sfruttando il file “ItalianTweetsMapping.json” disponibile nella cartella “elastic”.
- Aprire il progetto NiFi “IngesterGeolocatedItalianTweets”, configurare GetTwitter, quindi eseguire il progetto.
- Lasciatelo in Run così l’indice si comincia a riempire.