Social & Serie A TIM

Lorenzo Filippo Maria Colombo[1], Davide Pecchia[2], Michela Sessi[3]

[1] *Università degli studi di Milano Bicocca, Data Science; background: Informatica*
[2] *Università degli studi di Milano Bicocca, Data Science; background: Economia*
[3] *Università degli studi di Milano Bicocca, Data Science; background: Statistica*

Il calcio è lo sport nazionale in Italia ed è estremamente seguito dalla maggior parte della popolazione. Il campionato principale è la Serie A TIM, nella quale militano 20 squadre che nel corso della stagione si contendono il titolo finale. Le partite sono distribuite tendenzialmente nei giorni festivi, ad eccezione di alcuni turni infrasettimanali. In questo elaborato si è cercato di evidenziare, utilizzando Twitter come social network per reperire i dati, quali siano i fattori che maggiormente influenzano il traffico in rete. Per fare ciò si è utilizzata la Virtual Machine Azure (Microsoft) resaci disponibile dall'Università Bicocca, nella quale si è usufruito di Kafka per effettuare lo streaming dei tweets ed immagazzinarli in MongoDB (in formato .json). Dopodichè questi dati sono stati esportati in un nuovo database, Neo4j, nella quale sono stati aggregati con altre informazioni relative a squadre e partite (in formato .csv) in modo tale da poter successivamente effettuare le analisi ed ottenere i risultati desiderati.

1 Introduzione

La Serie A, inaugurata nel 1929, è il più ambito livello professionistico del campionato italiano di calcio. Dal 1997 la denominazione ufficiale è Serie A TIM, a seguito di un contratto di sponsorizzazione con la omonima compagnia telefonica. Per garantire l'alta prestazione e l'apertura del campionato, partecipano 20 squadre con 3 retrocessioni a fronte di 3 promozioni della serie B. Ogni stagione di campionato si compone di due gironi: uno di andata e uno di ritorno, in cui ogni squadra gioca una volta sola con tutte le altre. In totale quindi si svolgono 38 giornate di Serie A, ognuna comprendente 10 partite. Le squadre italiane non solo ambiscono a parteciparvi per il prestigio ed il profitto nazionale, ma anche per la popolarità internazionale: ad esempio, è attualmente al quarto posto nel ranking UEFA[1] per competizioni di club. La vittoria o una buona classificazione permette la qualificazione a grandi campionati quali UEFA Champions League[2] e UEFA Europa League[3]. Per la crescente popolarità negli anni ed il successivo sviluppo sempre più alto dei profitti derivanti dai diritti televisivi, dalle pubblicità e dal marketing delle squadre, le partite tendono ad essere giocate nel fine settimana così da garantirne maggiore visibilità ed accessibilità. Si tratta però di dieci partite che devono essere programmate dalla prima serata del Venerdì a quella del Lunedì. La scelta precisa del giorno e dell'orario contribuisce alla maggiore accessibilità, alla maggiore popolarità e al conseguente maggiore successo economico. Diversi sono i fattori che possono influenzare la popolarità di una partita e dunque il seguito in termini di visibilità: la classifica (una squadra di alta classifica sarà più seguita rispetto ad una di bassa classifica), l'orario, il numero di goal che vengono segnati, il giorno in cui viene disputata, ecc.
Per analizzare la popolarità ed individuare i fattori che maggiormente influiscono su di essa, si è utilizzato il social network Twitter, in cui ogni squadra possiede la propria pagina ufficiale e per ogni partita viene creato un hashtag ad hoc, in modo che le persone che la seguono possano manifestare le loro reazioni facendo riferimento a quest'ultimo. Con i dati raccolti è possibile determinare quali siano gli elementi che determinano le differenze di tweet (numericamente parlando) tra le partite di una classica giornata del campionato di Serie A.

2 Obiettivo

L'obiettivo della nostra ricerca è quello di evidenziare i fattori che influenzano il traffico sui social durante le partite della Serie A TIM. Ci siamo infatti chiesti se il numero di tweet relativi ad una partita (contenenti quindi l'hashtag ufficiale della partita stessa), dipendesse da elementi quali: il numero di followers di una squadra, ovvero il numero di persone che seguono la squadra in questione sul social network; la posizione in classifica, verificando se le squadre con un alto punteggio siano seguite di più rispetto a quelle con uno più basso, tendendo in considerazione un possibile scambio di posizioni rilevante per l'esito finale della competizione; l'orario in cui una partita viene giocata, verificando se partite disputate nei weekend o negli orari serali siano più seguite rispetto a partite di giorni feriali o in fascia pomeridiana; il numero di goal segnati, che potrebbero suscitare interesse rispetto ad una partita "piatta" e di conseguenza aumentare il traffico in rete.

[1] Sistema utilizzato dalla Union of European Football Association (UEFA) introdotto nel 1979 per classificare le squadre impegnate nei tornei continentali: le nazionali e i club.
[2] Il più prestigioso torneo internazionale calcistico in Europa per squadre di club maschili.
[3] Competizione sportiva, seconda per prestigio solo alla Champions League

3 Le venti squadre

Per ognuna delle venti squadre che militano nel campionato italiano di Serie A selezioniamo le seguenti variabili:

1. numero di followers di Twitter per la pagina ufficiale della squadra;
2. numero di post pubblicati su Twitter sulla pagina ufficiale della squadra;
3. spettatori medi allo stadio per ogni squadra (ultimo aggiornamento disponibile)[4]
4. ricavi per ogni squadra dai Main Sponsor della stagione corrente[5]
5. costo delle rose per squadra della stagione corrente.[6]

Da questi attributi creiamo un unico dataset `datisquadre.csv`.

4 Le dieci partite

Per ognuna delle dieci partite della ventiquattresima giornata del campionato selezioniamo le seguenti variabili:

1. hashtag ufficiale della partita reperito sulle pagine ufficiali delle squadre;
2. data della partita;
3. giorno della settimana in cui si disputa la partita;
4. orario in cui la partita viene giocata;
5. somma dei goal segnati durante il match;

Da questi attributi creiamo un unico dataset `partitegiornata24.csv`

5 Classifica settimanale

Ai fini dell'analisi anche la posizione in classifica di una squadra potrebbe risultare determinante e deve quindi essere tenuta in considerazione. Per ogni giornata scegliamo di ottenere la classifica precedente all'inizio delle partite. Si tratta di fatto di ottenere 20 posizioni (una per ogni squadra), lavoro che riuscirebbe facilmente anche manualmente. A puro scopo didattico e per curiosità abbiamo voluto usare WebScraper.io[7], estensione di Google Chrome che permette di fare scraping tramite la selezione di elementi all'interno di una pagina web. Abbiamo quindi costruito una Sitemap (una mappa della pagina nel nostro caso)dal sito della Gazzetta dello Sport[8] per permetterci di catturare i dati della tabella (alcuni anche superflui), impostando quale delle righe contenesse gli "headers" e quali invece contenessero le informazioni da scaricare. I 2 attributi di nostro interesse sono:

1. nome della squadra;
2. punteggio aggiornato.

La posizione non era riportata nella tabella in quanto le squadre erano semplicemente ordinate in base alla classifica. Necessitando del numero da poter inserire successivamente nel database, è stata inserita una ulteriore colonna, all'interno del file .csv, denominata "Posizione" Quindi, una

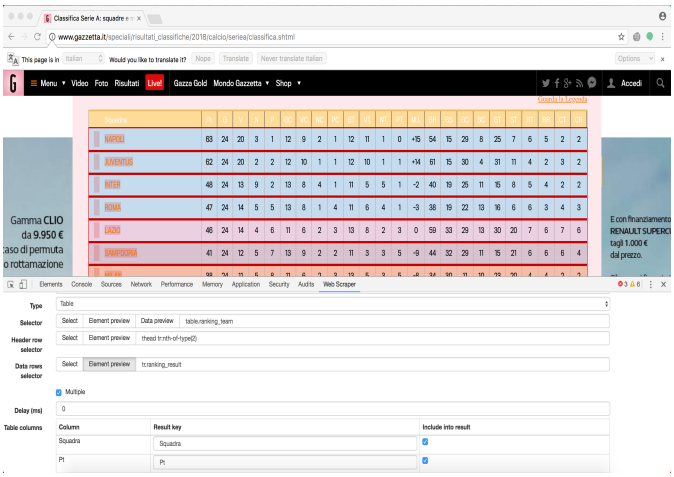

Figura 1: *Utilizzo di Webscraper.io per la classifica della serie A*

volta impostati i campi, questo strumento permette di ottenere il file completo `classificaserieagiornata24.csv` delle classifiche di ogni giornata che noi abbiamo scelto di scaricare ogni venerdì pomeriggio.

6 Twitter

Per svolgere le nostre analisi abbiamo pensato di far ricorso ad uno dei social network maggiormente utilizzato dai club della Serie A di calcio: Twitter. Infatti ogni squadra possiede un suo profilo ufficiale su cui vengono pubblicate le news, le comunicazioni, i risultati e gli aggiornamenti delle partite. Per accedere e poter quindi scaricare questi dati, il social rilascia delle API che permettono all'utente di "catturare" il 3% del traffico totale della rete in quel momento.
Facendo uso dell'architettura Kafka, abbiamo effettuato lo streaming dei tweets durante le partite in analisi (li abbiamo scaricati in real time) e immagazzinati in MongoDB per poter svolgere poi i passaggi successivi.

6.1 Architettura Kafka e MongoDB

Per garantire la velocità richiesta dal progetto ci siamo serviti, per il recupero dei dati, dell'ecosistema *Hadoop*, in particolare della piattaforma distribuita streaming Apache Kafka. Questa piattaforma si serve di:

- un *Producer*, per prendere i tweets in streaming e mandarli ad un topic creato appositamente dall'utente;
- un *Consumer* che, effettuando la sottoscrizione al topic creato, riesce a prendere in qualsiasi momento i dati immagazzinati;

[4] `https://www.panorama.it/sport/calcio/spettatori-serie-a-2017-2018-numeri-statistiche-squadre/`
[5] `https://uncalcioalleconomia.wordpress.com/2017/10/07/serie-a-201718-la-classifica-dei-ricavi-da-main-sponsor/comment-page-1/`
[6] `http://www.calcioefinanza.it/2018/01/18/stipendi-serie-a-2017-2018-e-costo-delle-rose/`
[7] `http://webscraper.io`
[8] `https://www.Gazzetta.it/speciali/risultati_classifiche/2018/calcio/seriea/classifica.shtml`

L'accesso alle API di Twitter è stato reso possibile da *tweepy*, una libreria python che permette di impostare il collegamento grazie a quattro token (consumer key, consumersecret, access token, access secret) resi disponibili direttamente da Twitter dopo aver creato un'applicazione.[9]
Per quanto riguarda lo storage dei dati recuperati dal *Consumer*, abbiamo pensato di immagazzinarli in MongoDB ogni volta che il *Consumer* riusciva a recuperare un tweet dal topic; questo per evitare perdite di tweets legate a malfunzionamenti e poter tener d'occhio informazioni come il numero di tweets scaricati o il volume raggiunto.
Per avere un arco temporale idoneo abbiamo scelto di scaricare i tweets per ogni partita a cominciare da 1h circa prima della data di programmazione e lasciare immagazzinare i dati per 4h così da prendere anche le reazioni alla fine della partita.

6.2 Giornata 21

Nella giornata 21 abbiamo sviluppato un codice che poi si è mostrato poco adatto al tipo di ricerca che volevamo fare, andando a prendere in formato stringa tutto il *massage.value* non differenziandone gli attributi al suo interno (testo, nome, ecc..). Abbiamo tuttavia deciso di includere questa raccolta nel progetto perchè ci interessava contare il numero di tweets raccolti e fare delle comparazioni nei risultati finali.

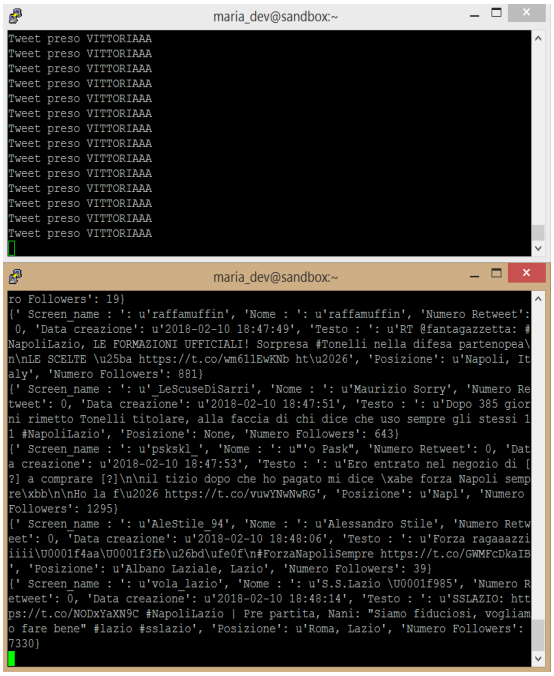

Figura 2: *Esempio di Streaming: Producer-Consumer*

6.3 Giornata 24

Dopo aver sperimentato diverse tipologie di codice per effettuare al meglio lo streaming dei tweets relativi alle partite, abbiamo ottenuto il codice migliore per raccogliere i dati finali nella ventiquattresima giornata di campionato. Ci siamo chiesti quale fosse il modo migliore per recuperare i tweets tra i due principali metodi resi disponibili da *tweepy*, ossia "on data" e "on status". La differenza principale tra i due è il numero di informazioni recuperate, molto di più per quanto riguarda il primo metodo, ma per la maggior parte inutili al fine del nostro progetto. Abbiamo quindi optato per il secondo metodo, andando a selezionare solo i pochi attributi necessari e rendendoli accessibili come parametri all'interno della lista passata al topic dal *Producer*. Altro passaggio importante è stato quello di mandare il dato al topic attraverso il comando *json.dumps*, per trasformare i dati da lista a stringa, e recuperarli nel *Consumer* attraverso una *json.load*, per riavere la nostra lista originale. Infine, con un ciclo FOR, abbiamo tenuto l'idea di immagazzinare tutti i tweets uno per uno su MongoDB e abbiamo stampato a video il tweet salvato, a scopo informativo.

6.4 Pulizia dei tweets

I tweets della giornata 24 sono stati scaricati in un unico topic, e trasferiti in un'unica collection in MongoDB; questo purtroppo non ci ha reso disponibile una immediata divisione dei tweets per partita. Attraverso alcune query su MongoDB abbiamo quindi cercato di suddividere i tweets associandoli alla relativa partita e hashtag di riferimeto. I tweets scaricati sono però sia tweets "ufficiali" quindi contenenti effettivamente l'hashtag della partita nel testo, sia retweets (RT) che talvolta contengono l'hashtag ed altre volte non lo contengono. Questo genera un problema di associazione. Abbiamo quindi provato a ricercare i tweets tramite espressione regolare, ad esempio:

```
db.collection.find ( {"Testo":{"$regex":
"RomaBenevento" , "$ options":"i"}} )
```

così da ottenere tutti i tweets che contengono nel testo l'hashtag, indipendentemente dalla scrittura maiuscola o minuscola. Come anticipato, in questo modo perdiamo i retweets che non hanno specificato l'hashtag: su 55677 tweets scaricati vengono ben classificati 42238, con una perdita di 13439. Per tentare un recupero di alcuni retweets proviamo a fare una ricerca separata, chiedendo che le due squadre giocanti la partita non vengano ricercate per hashtag ma separate, così da ottenere eventuali tweets che contengono il nome delle squadre in una frase:

```
db.collection.find ( {"Testo":{"$regex":  "Roma",
"$regex":  "Benevento", "$ options":"i"}} )
```

questa ricerca classifica 48011 tweets, diminuendo la perdita a 7666. Sembrerebbe un ottimo risultato se non fosse che in questo modo abbiamo degli errori di classificazione. Potrebbe infatti esistere un tweet riferito ad un certo hashtag che viene però classificato sotto un'altra partita perchè contenente il nome delle squadre di quest'ultima. Inoltre alcuni tweets per lo stesso motivo vengono assegnati a più cluster perchè soddisfacenti entrambi i requisiti di ricerca. Un chiaro esempio di questo

[9] `https://apps.twitter.com`

problema è proposto nella figura 3. Per questo motivo

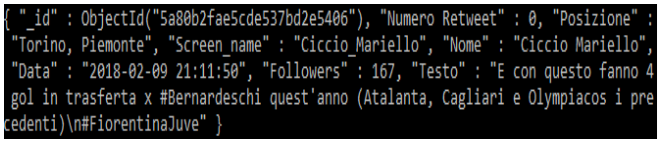

Figura 3: *Esempio di misclassificazione: tweet scaricato dall'hashtag #FiorentinaJuve classificato male per la presenza della parola Atalanta*

abbiamo scelto di considerare solo i tweets che ufficialmente citavano la partita tramite hashtag, accontentandoci di una perdita di informazioni.
Abbiamo creato nuove collections nominate con le iniziali delle squadre giocanti (squadra in casa- squadra ospite in MongoDB una per ogni partita che racchiuda i tweets associati così da facilitarci nell'esportazione dei dati in formato *json*. Un esempio di creazione è il seguente:

```
db.collection.aggregate ( [ {$match:  {"Testo":
{"$regex":  "RomaBenevento", "$options":"i"}}} ,
{$out :  "RB"} ] )
```

Otteniamo quindi 10 file distinti (ad esempio `RB.json`, `IB.json`, `CA.json`) in formato .json, uno per ogni partita associato all'hashtag corrispondente.

7 Aggregazione

A questo punto abbiamo una serie di file di formati differenti:

1. `datisquadre.csv`;
2. `partitegiornate24.csv`;
3. `classificaserieagiornata24.csv`;
4. dieci file contenenti i tweets di ogni partita: `FJ.json`, `SM.json`, `CA.json`, `NL.json`, `SC.json`, `SV.json`, `IB.json`, `CG.json`, `TU.json`, `RB.json`.

Abbiamo escluso l'idea di inserire tutte le informazioni in un DB document-based, nel nostro caso in MongoDB, perchè creava contrasti ideali. Volevamo infatti cercare di costruire un DB scalabile nell'idea di potere inserire i dati di tutte le giornate del campionato. In un documento esplicitiamo cosa riteniamo più importante tramite l'ID che diventa la chiave primaria per il nostro DB. È una struttura molto efficiente se vogliamo tutte le informazioni di un certo oggetto (ID) ma se dobbiamo allineare diverse richieste contrastanti diventa scomodo. Inoltre, nel nostro caso, avrebbe sicuramente portato ad una ripetizione dei dati, situazione da evitare per non insorgere in errori soprattutto per eventuali aggiornamenti.
Ad esempio, se avessimo pensato di dividere i documenti in base alle partite come chiave primaria, dove avremmo inserito i dati delle singole squadre? Avremmo dovuto ripeterli per ogni diversa partita giocata in una giornata differente. Al contrario se avessimo impostato il nostro DB in base alle squadre avremmo dovuto ripetere i dati delle partite (e ovviamente tutti i tweets) due volte, poiché ovviamente una partita presuppone lo scontro di due avversari. Incorreremmo inoltre in problemi computazionali avendo richieste con differenti punti di vista.

7.1 Neo4j

Per aggregare tutte queste informazioni abbiamo scelto di utilizzare Neo4j[10]. Il supporto di un modello a grafo infatti ci permette di collegare tramite arco (relazione) le squadre alle partite senza dover duplicare informazioni. Nell'idea di potere inserire non solo l'intero campionato, ma anche un'analisi dell'andamento sella Serie A TIM negli anni è da constatare che Neo4j non supporta lo sharding dei dati, quindi l'intero grafo deve essere memorizzato in una sola macchina. Nonostante ciò è in grado di memorizzare decine di miliardi di nodi.

7.2 Costruzione del graphDB

Per formare il nostro grafo procediamo con la creazione dei nodi delle squadre con i relativi attributi già puliti del file `datisquadre.csv`:

```
LOAD CSV WITH HEADERS FROM
"file:/.../datisquadre.csv" as datisquadre
CREATE (s:squadra {nome:datisquadre.Squadra,
followers:datisquadre.Followers,
tweets:datisquadre.Tweets,
spettatori_medi:datisquadre.Spettatori,
ricavi_main_sponsor:datisquadre.MainSponsor,
valore_rosa:datisquadre.ValoreRose })
```

Dal file delle partite creiamo i dieci nodi:

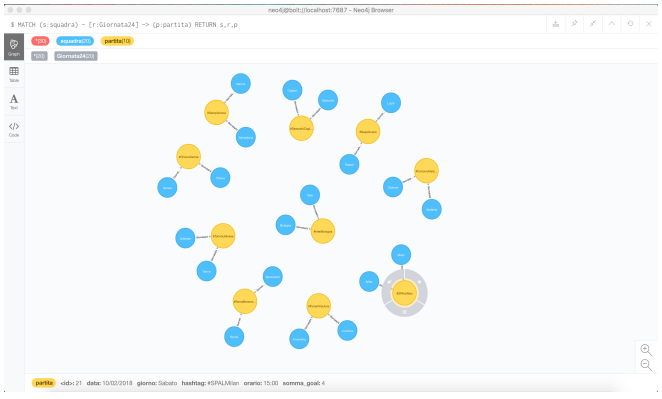

Figura 4: *Partitite, squadre e relazioni per la giornata 24*

```
LOAD CSV WITH HEADERS FROM
"file:/.../partiteaggiornate24.csv" as
datipartite
CREATE (p:partita {hashtag:datipartite.Partita,
data:datipartite.Data, orario:datipartite.Orario,
giorno:datipartite.Giorno,
somma_goal:datipartite.SommaGoal })
```

[10] NoSQL graph DB rilasciato dal 2007 e sviluppato dalla Neo Technology, una startup di Malmö, Svezia e della San Francisco Bay Area.

Si creano quindi le relazioni (Giornata24) tra rispettive squadre e partite con l'inserimento di due nuovi attributi: la posizione in classifica prima della giocata e il risultato della partita (vittoria o sconfitta).
Per l'importazione dei file .json abbiamo istallato la libreria APOC e impostato sia i nodi dei tweets che le relazioni con la partita. Un esempio per la partita Fiorentina-Juventus:

```
CALL APOC.LOAD.JSON("file:/.../FJ.json")as FJ
CREATE (t:tweet {screen_name:FJ.Screen_name,
nome_utente:FJ.Nome, data_e_ora:FJ.Data,
followers:FJ.Followers, Posizione:FJ.Posizione
})
WITH t
MATCH (p:partita {hashtag:  "#FiorentinaJuve"}
CREATE (p) - [a:tweet] -> (t)
```

Otteniamo quindi in totale 42268 nodi e 42258 relazioni.

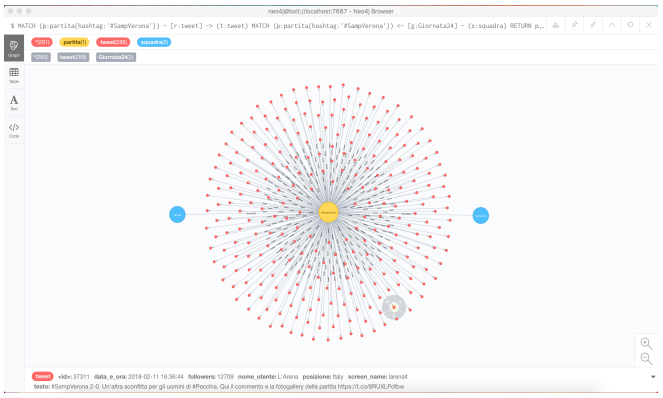

Figura 5: *Esempio di query: tweets e squadre relazionate al nodo della partita SampVerona*

8 Alcune analisi e risultati

Per problemi di connessione abbiamo purtroppo perso i tweets nella fascia oraria 15.20-16.20 di domenica ottenendo quindi un buco di circa un'ora. Per cercare di confrontare i conteggi finali stimiamo quindi un andamento costante al solo fine di ottenere una valutazione arbitraria per le quattro partite giocate alle 15.00. Il nuovo conteggio aggiunge quindi un terzo dei risultati ottenuti: $totale\_finale = totale + totale/3$. Dai dati analizzati sono emersi risultati interessanti in diversi ambiti: innanzitutto è evidente come il traffico di tweets si distribuisca in modo discontinuo a seconda delle partite (figura 6). Le partite programmate per le 20.45, prime serate televisive, sembrano essere quelle che acquistano maggiore visibilità. In ogni caso la terza posizione la conquista la partita Inter-Bologna che è stata giocata di domenica pomeriggio in concomitanza con altre tre partite. Aldilà degli orari infatti, sembra apparire più significativa la posizione in classifica: le partite che contano più tweets sono quelle giocate dal secondo posto (Juventus), dal primo e terzo posto (Napoli-Lazio) e dal quarto posto (Inter).

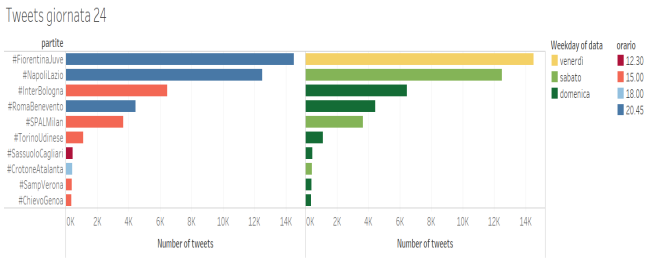

Figura 6: *Giornata 24*

8.1 Confronto Fiorentina-Juve/Napoli-Lazio

Nonostante la posizione in classifica la partita Napoli-Lazio ottiene meno tweets della partita Fiorentina-Juventus. Si osserva dalla figura 7 come i picchi di traffico siano avvenuti durante i goal della Juventus e nel momento in cui si è ricorso al VAR (Video Assistant Referee, introdotto in questa stagione calcistica e sempre più discusso di partita in partita).

Dalla figura 8 vediamo invece come tale partita sia stata molto più attiva: i goal totali sono cinque. Sebbene un numero maggiore di goal e di altri eventi porterebbe a far pensare ad un aumento di traffico sui social non sembra esserci una dimostrazione a favore della maggiore pubblicazione di tweets nella partita Fiorentina-Juventus in questo senso.

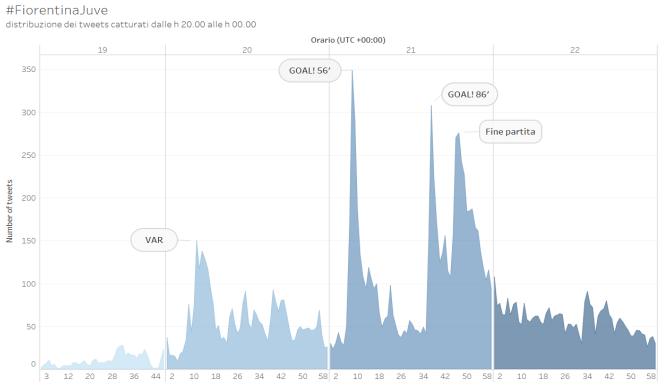

Figura 7: *Distribuzione dei tweets per la partita Fiorentina-Juventus*

È da valutare tuttavia un attributo fondamentale: quanto

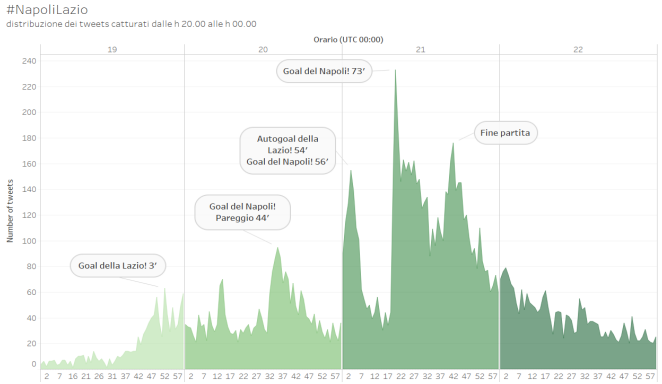

Figura 8: *Distribuzione dei tweets per la partita Napoli-Lazio*

effettivamente la squadra sia seguita su Twitter. Emerge nella figura 9 l'assoluto distacco numerico tra Milan e Juventus rispetto a tutte le altre squadre. Il numero di followers non sono quindi ininfluenti ma al contrario sembrano essere per questa partita una delle variabili più correlate (per il caso specifico).

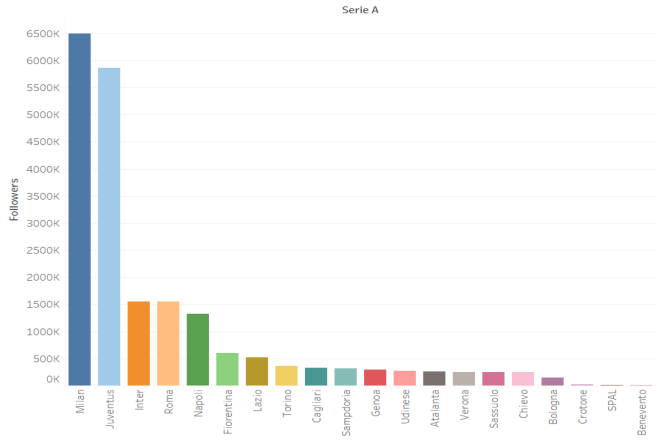

Figura 9: *Numero di Followers su Twitter della pagina ufficiale della squadra*

8.2 Confronto con la giornata 21

Come abbiamo anticipato, per la giornata 21 abbiamo ottenuto esclusivamente un conteggio totale. In particolare catturiamo 34095 tweets (comprendenti anche i retweets), numero che risulta decisamente inferiore rispetto ai 55677 della giornata 24. Si può notare come nella giornata 21 si

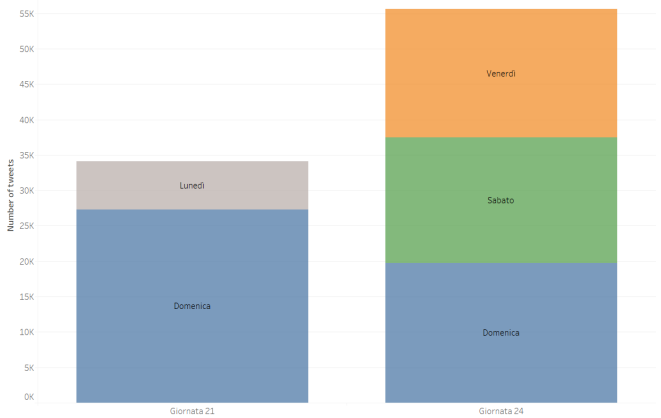

Figura 10: *Traffico dei tweets e visione dei giorni della settimana per le giornate 21 e 24*

siano giocate tutte le partite tra Domenica e Lunedì, mentre nella giornata 24 le partite sono spalmate su 3 giorni, da Venerdì a Domenica. Il numero superiore di tweets della ventiquattresima giornata è dunque dovuto, come si evince dal grafico in figura 10, ai giorni in cui vengono disputate le partite: infatti 3 prime serate nella giornata 24 (una Venerdì, una Sabato ed una Domenica) hanno fatto sì che il traffico fosse superiore rispetto alla giornata 21, nella quale si sono giocate 2 partite alle 20.45 (di cui una però in un giorno lavorativo). Osservando in particolare la parte del grafico relativa alla sola giornata 24, emerge anche come la squadra che disputa il match sia determinante in termini di numero di tweet: infatti il Venerdì si è giocata soltanto una partita (Fiorentina-Juventus) ed il numero di tweets ha eguagliato quello dei giorni successivi in cui hanno giocato tutte le altre squadre. Ricordiamo che la Juventus è la squadra numero 2 della classifica.

9 Criticità e conclusioni

Le criticità principali sono state riscontrate nell'ambito dello sviluppo del codice per lo streaming dei tweets, che ha richiesto più passaggi per essere perfezionato ma che alla fine è stato sviluppato correttamente ed i dati ottenuti sono stati soddisfacenti in termini di qualità. Si sono anche verificati alcuni imprevisti durante l'utilizzo delle macchine virtuali che, causa o interruzioni della connessione internet o problemi di connessione alla macchina stessa, hanno momentaneamente interrotto lo streaming in qualche occasione; le perdite di informazione in ogni modo, data la quantità di tweet raccolti, non sono state significative ai fini della ricerca. Queste sono state le principali criticità incontrate.
Dalla ricerca effettuata è dunque emerso come il traffico di Twitter sia fortemente influenzato innanzitutto dall'andamento della partita (si twitta di più quando succede qualcosa di interessante come un goal o un fallo controverso); in secondo luogo, la posizione in classifica suscita maggior interesse nei confronti della squadra: si è infatti riscontrato che le 2 partite col maggior numero di tweets siano quelle in cui erano impegnate le squadre in testa alla classifica, ovvero Juventus e Napoli; si nota anche come il numero di followers di una squadra non influenzi particolarmente il numero di tweets nella maggior parte dei casi, inducendo a concludere che sia più la partita in sè che suscita interesse (una partita più interessante ai fini della classifica crea maggior traffico a livello di tweets piuttosto che una partita in cui è impegnata una squadra molto seguita su Twitter ma più in basso nel ranking; esempio: SPAL -Milan, ricordando che il Milan ha il numero maggiore di followers rispetto a tutte le altre sqaudre). Infine si può constatare come distribuire le partite in modo da garantire più prime serate in giorni festivi generi più traffico rispetto a partite collocate in giorni lavorativi in quanto la gente è più propensa a seguirle e di conseguenza, chi lo desidera, a twittare.
Per migliorare i risultati ottenuti ed averne di maggiormente accurati sarebbe possibile ottenere i dati di ulteriori giornate in modo da svolgere le analisi su una maggiore quantità di informazioni, il che è facilmente realizzabile visto l'ottima scalabilità del database da noi utilizzato.