Progetto per il corso di Data Management

Diego Bartoli Geijo, Lorenzo Bruni
Università Milano Bicocca

Dipartimento di Informatica Sistemistica e Comunicazione
Corso di laurea magistrale in Data Science

Gennaio 2022

1 Introduzione

É possibile individuare una fascia di età in cui è probabile che un calciatore raggiunga il suo massimo rendimento basandosi sulla nazionalità del calciatore, le caratteristiche fisiche, il ruolo o il campionato in cui gioca? Si vuole costruire un dataset che possa servire come riferimento per cercare di rispondere a tali domande.

2 Data Acquisition

Dopo un'accurata ricerca selezioniamo tre fonti dati: Understat[5], api-football[1], Fbref[3] . Analizzati i dati disponibili scegliamo di coprire le stagioni dalla 2017/2018 alla 2020/2021 dei 5 principali campionati europei: Premier League (Inghilterra e Galles), Serie A (Italia), La Liga (Spagna), Bundesliga (Germania) e Ligue 1 (Francia).

Da Understat otteniamo le statistiche offensive dei calciatori, da api-football otteniamo informazioni anagrafiche e fisiche sui calciatori, da Fbref otteniamo statistiche difensive e dei passaggi dei calciatori. Scarichiamo soltanto i dati dei giocatori con più di 15 partite nella stagione considerata perché riteniamo non valutabile una stagione con meno di 15 partite giocate. Da Understat e api-football otteniamo un file json per ogni stagione di un certo campionato, quindi 20 file da ciascuna (5 campionati, 4 stagioni). Da Fbref otteniamo due file json per ogni stagione di un certo campionato, uno per le statistiche difensive e uno per le statistiche dei passaggi, quindi 40 file. Otteniamo in totale 80 file. Ogni file contiene un array di object json, un object per giocatore. Per una descrizione dettagliata del processo di data acquisition si rimanda il lettore al Google Colab.

3 Data Modelling

I dati scaricati da Understat e da api-football sono forniti sotto forma di dizionari python, un dizionario per giocatore. Salviamo i dati in file json, poiché ci sembra il formato più adatto per gestire tale struttura. I dati raccolti da Fbref sono forniti in formato tabellare ma li trasformiamo in modo da ottenere anche in questo caso dei dizionari python per omologare gli schemi e per poter salvare anche questi in file json. Per dettagli sul procedimento si rimanda il lettore al Google Colab, sezione Fbref.

Vista la struttura dei dati a disposizione il modello NoSQL più adatto per il nostro database è il document-based. Come document-based management system scegliamo MongoDB, per preferenze personali e perché è uno di quelli di più diffuso utilizzo. Analizzati i dati a disposizione stabiliamo la seguente struttura per il nostro database:

- 1 collezione per ogni campionato e stagione dati
- 1 documento per ogni giocatore di una certa collezione

- Schema dei documenti, i dati sono riferiti alla stagione considerata:
  - name: nome
  - age: età
  - nationality: nazionalità principale
  - height: altezza in cm
  - weight: peso in kg
  - team: squadra di appartenenza
  - position: posizione principale, i valori possibili sono FW(attaccante), MF(centrocampista), DF(difensore), GK(portiere)
  - general stats
    - games: numero di partite giocate
    - time: minutaggio totale
    - red cards: numero di cartellini rossi ottenuti
    - yellow cards: numero di cartellini gialli ottenuti
  - offensive stats
    - goals: numero di goal segnati
    - xG: expected goals[4] generati
    - assists: numero di assist forniti
    - xA: expected assists[2] forniti
    - shots: numero di tiri realizzati
    - key passes: numero di passaggi che hanno portato il ricevente al tiro ma non al goal realizzati
    - npg: numero di goal senza contare i rigori realizzati
    - npxG: expected goals generati senza contare i rigori
    - xGChain: expected goals totali di ogni possesso in cui il giocatore è stato coinvolto
    - xGBuildup: expected goals totali di ogni possesso in cui il giocatore è stato coinvolto esclusi key passes e tiri
  - defensive stats
    - Tkl: numero di tackles realizzati
    - TklW: numero di tackles che hanno portato al recupero del pallone realizzati
    - Past: numero di volte che il giocatore è stato dribblato
    - Press: numero di pressing sull'avversario realizzati
    - Succ: numero di volte che è stato recuperato il possesso del pallone meno di 5 secondi dopo l'inizio dell'azione di pressing
    - Blocks: numero di volte che il giocatore ha recuperato il pallone
    - Int: numero di anticipi realizzati
  - passing stats
    - Cmp: numero di passaggi completati
    - Cmp%: percentuale di passaggi completati sui passaggi totali
    - 1/3: numero di passaggi verso il terzo di campo finale in attaco esclusi calci piazzati completati
    - PPA: numero di passaggi verso l'area di rigore esclusi calci piazzati completati
    - CrsPA: numero di cross verso l'area di rigore esclusi calci piazzati completati
    - Prog: numero di passaggi, che si muovono di almeno 9 metri verso la porta avversaria dal punto più lontano da essa negli ultimi sei passaggi o sono verso l'area di rigore, completati

4 Data cleaning

Sono necessarie alcune operazioni di cleaning dei dati. Riportiamo di seguito le operazioni eseguite, divise per fonte dei dati. Per dettagli sull'implementazione di tali operazioni si rimanda il lettore al Google Colab.

Understat I valori di alcuni campi che logicamente dovrebbero essere numerici, come ad esempio il numero di gol, vengono forniti in formato di stringa. Individuiamo i campi che presentano tale problematica e ne convertiamo opportunamente il tipo dei valori. Alcuni giocatori presentano un doppio valore per il campo squadra, sono giocatori che hanno cambiato squadra a metà stagione restando nello stesso campionato. Normalizziamo il valore del campo squadra dove necessario e manteniamo soltanto il primo valore. Le statistiche sono fornite aggregate tra le due squadre quindi perdiamo informazione ma la riteniamo un'approssimazione accettabile poiché per lo scopo per cui è pensato questo dataset la squadra di appartenenza non è particolarmente importante, è più importante il campionato. Cercare di individuare un'età di rendimento massimo all'interno di una squadra infatti non ha senso, perché il numero di giocatori di una squadra è troppo piccolo per essere statisticamente significativo. Inoltre i giocatori che presentano due valori per il campo squadra sono soltanto lo 0.03% dei giocatori totali considerati.

Api-football I valori di peso e altezza dei giocatori sono forniti in formato di stringa. Noi riteniamo che siano più fruibili come interi quindi li convertiamo. Il peso è fornito in kg e l'altezza in cm. Nei dati viene fornita soltanto l'età attuale dei giocatori mentre a noi interessa ottenere l'età dei giocatori nella stagione considerata, per poter associare età e rendimento. Utilizzando la data di nascita, fornita nei dati, ricaviamo opportunamente l'età.

Fbref Anche in questo caso valori logicamente numerici vengono forniti in formato di stringa, quindi li convertiamo opportunamente. Risulta inoltre necessaria un'operazione di normalizzazione dei valori del campo che indica il ruolo del giocatore, per alcuni giocatori ne è infatti indicato più di uno. Non conserviamo tutti i valori poiché riteniamo che per lo scopo per cui è pensato questo dataset sia più adatto che un giocatore abbia soltanto un ruolo. A nostro avviso il ruolo sarà un fattore importante nel determinare l'età di rendimento massimo di un giocatore e la presenza in più ruoli di diversi giocatori potrebbe difficoltare l'individuazione di pattern di rendimento. Potremmo introdurre nuovi ruoli, ad esempio un giocatore con ruolo centrocampista e attaccante potrebbe diventare un centrocampista offensivo ma rischieremmo di diminuire in eccesso il numero di giocatori per ogni ruolo. Inoltre da un'analisi di un campione dei dati in base alle nostre conoscenze personali riteniamo che in molti casi l'indicazione di due ruoli sia abbastanza forzata. Conserviamo soltanto il primo valore ritendendolo il ruolo principale del calciatore. I ruoli possibili sono GK, DF, MF e FW. Potremmo standardizzarli trasformandoli in goalkeeper, defender, midfielder e forward ma riteniamo non sia necessario perchè sono abbreviazioni standard e di uso molto diffuso.

5 Data integration

In fase di download dei dati imponiamo che stessi giocatori abbiano lo stesso nome in tutti i dataset. Per i dati scaricati da Fbref e api-football, sostituiamo il nome di ogni giocatore considerato con il nome con cui ha la massima somiglianza tra quelli dei giocatori scaricati da Understat. Se per un certo nome non ne viene trovato nessuno con valore di somiglianza superiore ad una soglia stabilita il giocatore viene scartato. Per dettagli sul procedimento si rimanda il lettore al Google Colab. La corrispondenza che rende possibile l'aggregazione dei dataset è quindi tra i campi contenenti i nomi dei giocatori.

Tutti i dati scaricati hanno la struttura di object json quindi non abbiamo conflitti di schema. Abbiamo un instance level conflict per quanto riguarda il ruolo dei giocatori. Tale campo è presente sia nei dati scaricati da Understat che in quelli scaricati da Fbref e i formati dei valori sono diversi. Dopo un'analisi a campione dei valori decidiamo di mantenere soltanto quelli di Fbref perché di più facile interpretazione.

Abbiamo 80 file ottenuti da 3 sorgenti dati diverse. Analizzandone i contenuti osserviamo che i file scaricati da api-football contengono generalmente meno giocatori. Questo è dovuto alla maggior differenza nei formati dei nomi tra i dati di Understat e api-football rispetto ai dati di Understat e Fbref. Decidiamo quindi di prendere come riferimento i giocatori presenti nei file scaricati da api-football e aggregare a questi le statistiche offensive, difensive e di passaggi.

6 Data quality

Accuracy Dal punto di vista della syntactic accuracy possiamo analizzare i campi con valori in formato di stringa, cioè il nome dei giocatori, il ruolo e la nazionalità. I nomi con cui alcuni giocatori sono conosciuti possono variare leggermente da fonte a fonte quindi è difficile stabilire dei valori rispetto ai quali applicare una funzione di distanza per valutare l'accuracy dei valori del dataset. Noi abbiamo deciso di utilizzare i nomi forniti da Understat perché ad un'analisi a campione ci sembravano fruibili e di uso comune. Anche per i valori di ruolo e nazionalità sono stati adottati dei formati di facile comprensione. Per quanto riguarda la semantic accuracy il problema è la difficile definizione di corretta rappresentazione dei valori nel mondo reale. Nel nostro caso è difficile stabilire universalmente quali siano le metriche necessarie per calcolare al meglio il rendimento di un giocatore. Noi possiamo però affermare che nel nostro database sono presenti metriche moderne e specifiche per le tre fasi principali del gioco, offensiva, difensiva e di possesso. Riteniamo quindi di fornire le sufficienti informazioni per valutare il rendimento di un calciatore.

Completeness Per quanto riguarda l'analisi dei valori nulli o mancanti nel nostro dataset circa lo 0.1% dei giocatori non possiede le informazioni complete. Sono i giocatori presenti nei dati scaricati da api-football ma non in quelli scaricati da Fbref, non presentano quindi defensive stats e passing stats. Passiamo ora a studiare la object completeness. Ponendoci in un ipotesi di mondo aperto vogliamo stimare quanti oggetti rappresentabili non sono presenti nel nostro database, quindi quanti giocatori con più di 15 partite in una certa stagione di un campionato non sono presenti nell'opportuna collezione. Nel nostro database non sono presenti tutti i giocatori inizialmente scaricati da Understat, per le ragioni spiegate nella sezione Data Integration del presente documento. Assumiamo che i file scaricati da Understat contengano effettivamente tutti i giocatori rappresentabili. Andiamo quindi a calcolare per ogni stagione e anno il rapporto tra il numero di oggetti nella collezione del nostro database e il numero di giocatori nel rispettivo file Understat. Riportiamo di seguito i risultati per ogni stagione e campionato.

Collezione	Completezza
Premier League 2017	0.90
Serie A 2017	0.92
La Liga 2018	0.88
Bundesliga 2018	0.95
Ligue 1 2018	0.88
Premier League 2018	0.93
Serie A 2018	0.99
La Liga 2018	0.93
Bundesliga 2018	0.99
Ligue 1 2018	0.91
Premier League 2019	0.94
Serie A 2019	0.92
La Liga 2019	0.86
Bundesliga 2019	0.93
Ligue 1 2019	0.91
Premier League 2020	0.92
Serie A 2020	0.94
La Liga 2020	0.91
Bundesliga 2020	0.95
Ligue 1 2020	0.93

La completezza media è pari al 92%.

Consistency Analizziamo la consistenza delle diverse rappresentazioni di uno stesso giocatore nel database. I dati forniti da Understat sono consistenti. Il nome di ogni giocatore resta uguale tra i vari dataset forniti. Come spiegato nella sezione Data Integration, noi abbiamo deciso di utilizzare come riferimento i nomi dei giocatori scaricati da Understat. Anche a costo di perdere qualche giocatore abbiamo imposto che i nomi dei giocatori del nostro dataset fossero uguali a quelli di Understat. Abbiamo quindi scelto di perdere in completeness per massimizzare la consistency. In questo modo possiamo essere sicuri che il nome di un giocatore è uguale in tutti i documenti del dataset che lo rappresentano.

Riferimenti bibliografici

[1] api-football. https://www.api-football.com/.

[2] expected assists. https://shortest.link/2BS0.

[3] Fbref. https://fbref.com/en/.

[4] expected goals. https://en.wikipedia.org/wiki/Expected_goals.

[5] Understat. https://understat.com/.